E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Diseño y construcción de un head-tracker para aplicaciones de sonido inmersivo

Grado en Ingeniería
en Tecnologías de Telecomunicación

Trabajo Fin de Grado

Autor: Sergio Castillo

Tutores: Ricardo San Martín, Asier Marzo

Pamplona, junio de 2020

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

ÍNDICE

INTRODUCCIÓN Y OBJETO

El trabajo fin de grado presente consiste en el desarrollo de un sistema Head-Tracker de bajo coste. Estos sistemas determinan la posición y orientación de la cabeza del usuario, facilitando la interactividad en aplicaciones de realidad virtual.

En experiencias inmersivas sonoras presentadas por medio de auriculares, monitorizar la posición de la cabeza del oyente supone un incremento notable de la calidad percibida. Los sistemas comerciales que lo implementan suelen ser costosos y estar asociados a aplicaciones propietarias.

Mediante elementos propios de la cultura maker (plataforma Arduino, impresión 3D, open sound control OSC...) se diseñará y construirá un dispositivo que comunicará la posición de la cabeza (yaw, pitch, roll) mediante bluetooth para ser combinado con plugins y aplicaciones de código abierto.

Con todo ello se pretende encontrar una alternativa a este tipo de tecnologías más asequible y de fácil implementación.

1- PROYECTO SONORIZARTE

Este proyecto fin de grado se enmarca en otro de mayor envergadura, Sonorizarte. El proyecto consiste en la creación de una esfera de 24 altavoces para la creación de sonido inmersivo sobre la que se proyectarán imágenes para crear un entorno virtual (ver Figura 1). Gracias a la llegada de la realidad virtual conocemos la posibilidad de interactuar con imágenes en 3D. Sin embargo, también es posible la interacción con sonido tridimensional. Esta instalación permite experimentar con escenas sonoras de forma totalmente inmersiva.

Este proyecto propondrá a artistas de diferentes disciplinas la creación de una instalación sonora conjunta donde puedan experimentarse escenas sonoras en audio 3D. La instalación podrá funcionar en modo exposición, mostrando creaciones realizadas por distintos compositores; o en modo interactivo, en el que el público crea en tiempo real una determinada escena o paisaje sonoro.

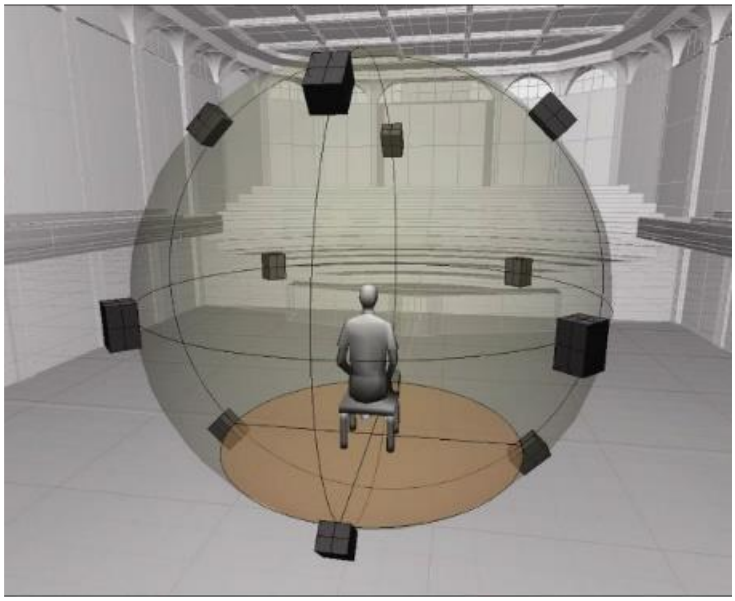


FIGURA 1: Idea de la instalación

Desde el punto de vista acústico se dirigirá la implementación de las interfaces para la renderización y la interacción con el usuario, los algoritmos de presentación específicos requeridos para la instalación y las necesidades técnicas.

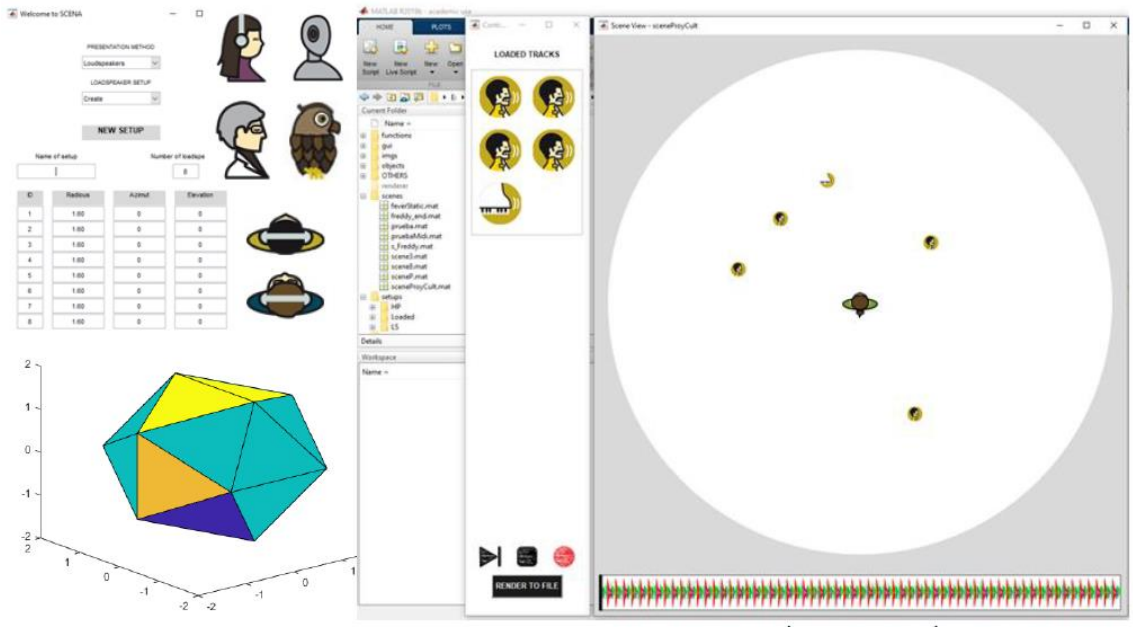


FIGURA 2: Idea de los interfaces

Asimismo, la instalación expondrá escenas sonoras creadas utilizando tanto sonidos de la naturaleza como de entornos del patrimonio cultural y arquitectónico, lo que permitirá la recreación inmersiva de paisajes naturales e idiosincráticos y el análisis auditivo de los mismos.

El dispositivo Head-Tracker permitirá simular el resultado de la esfera de manera virtual. Tiene la capacidad de ofrecer al usuario la rotación de la escena sonora para una presentación binaural en auriculares.

2- HEAD-TRACKING PARA LA MEJORA DE EXPERIENCIAS INMERSIVAS

2.1- Tecnología Binaural

La tecnología binaural es un método de grabación y presentación de sonido cuyo objetivo es registrar y reproducir el campo sonoro tal y como se representa en la entrada de nuestros oídos. De esa manera, pretende recrear, la sensación de sonido de un oyente como si éste se encontrara inmerso en la escena sonora.

La capacidad del ser humano para localizar el sonido 3D es importante pues le advierte del peligro y ayuda a separar los sonidos individuales del conjunto de sonidos del mundo acústico que lo rodea.

En este trabajo se utiliza un sistema de referencia definido por 3 variables: distancia (r), azimut (φ) y elevación (δ).

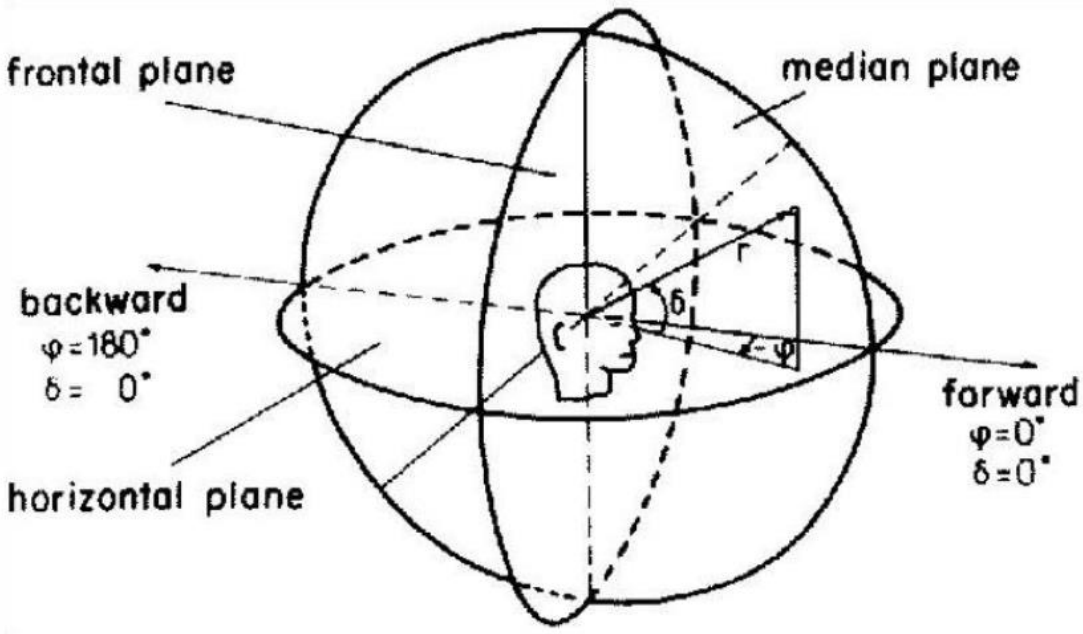


FIGURA 3: Sistema de referencia

Si una fuente de sonido se encuentra a la derecha de la dirección hacia adelante del oyente, entonces el oído izquierdo está en la sombra proyectada por la cabeza del oyente. Por lo tanto, la señal en el oído derecho debería ser más intensa que la señal en el oído izquierdo. Ésta diferencia de nivel es una referencia importante de que la fuente de sonido se encuentra en el lado derecho.

La comparación estándar entre intensidades en los oídos izquierdo y derecho se conoce como la **diferencia de nivel interaural (Interaural Level Difference, ILD)**.

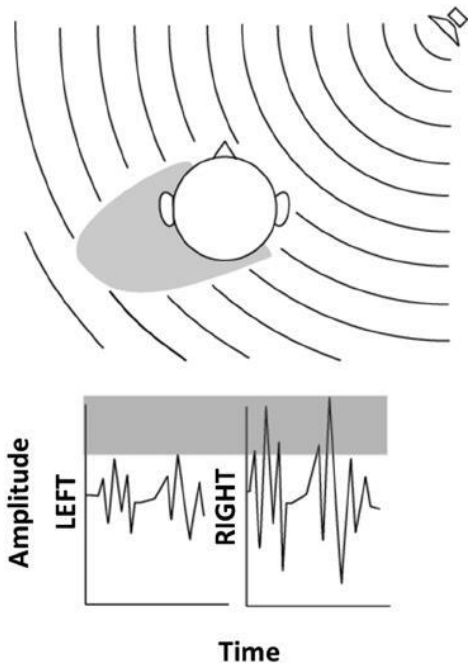


FIGURA 4: Interaural Level Difference

La diferencia de nivel se expresa en decibelios y se mide dentro del mismo período de tiempo para los dos oídos. Ésta diferencia de nivel depende de la longitud de onda de las señales de sonido que se reciben, para una longitud de onda mayor se produce una mayor difracción.

$$\lambda = \frac{c}{f} \tag{1}$$

A una frecuencia de 500Hz la longitud de onda del sonido es de 69cm, cuatro veces el diámetro de la cabeza humana (promedio). Por lo tanto, el ILD es pequeño para frecuencias inferiores a 500Hz. Pero la dispersión por la cabeza aumenta rápidamente con el aumento de la frecuencia y a partir de los 4.000Hz la cabeza proyecta una sombra significativa. Para todos estos supuestos hay que tener en cuenta que la fuente de sonido se encuentre a más de 1m de distancia. (ver Figura 5)

También cabe destacar que la ILD depende de la sensibilidad del sistema nervioso a las diferencias de nivel. Para el ser humano el cambio más pequeño detectable es de 0,5dB.

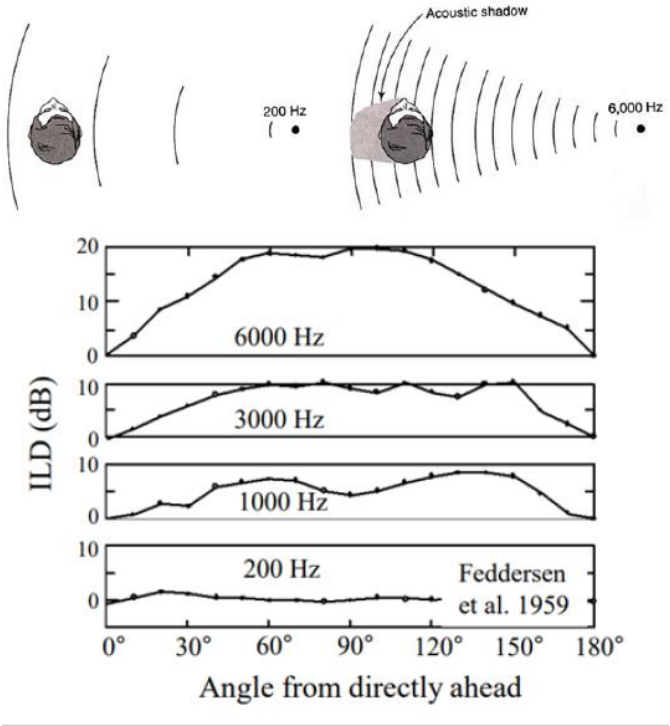


FIGURA 5: ILD

Por otro lado, cuando una señal llega a la cabeza del oyente desde un lado, la señal tiene que viajar más lejos para llegar al oído lejano que al oído cercano. Esta diferencia de longitud de camino da como resultado una diferencia de tiempo entre las llegadas del sonido a los oídos, que se detecta y también ayuda al proceso de localización de la fuente de sonido. A este fenómeno se le conoce como **diferencia de tiempo interaural (Interaural Time Difference, ITD)**.

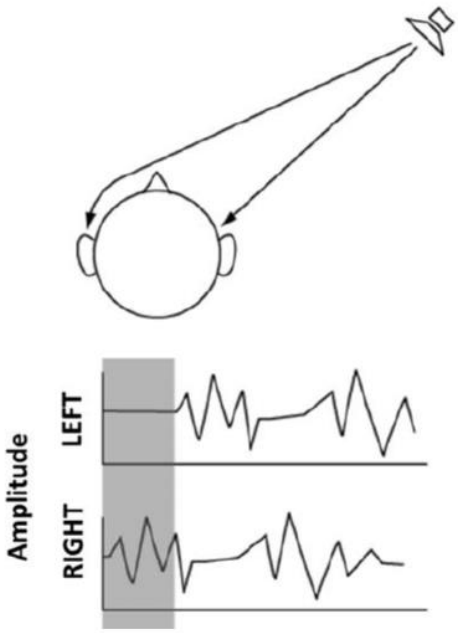


FIGURA 6: Interaural Time Difference

Un modelo simplificado para calcular la ITD en función del azimut (φ) es:

$$\mathrm{ITD} = a\,\sin(\varphi) \tag{2}$$

Donde a es el radio de la cabeza del oyente. También se puede calcular como diferencia de caminos, esto se refiere a que las ondas que llegan a los dos oídos se pueden desplazar en el tiempo hasta que coincidan perfectamente, el tamaño del desplazamiento será el valor de ITD.

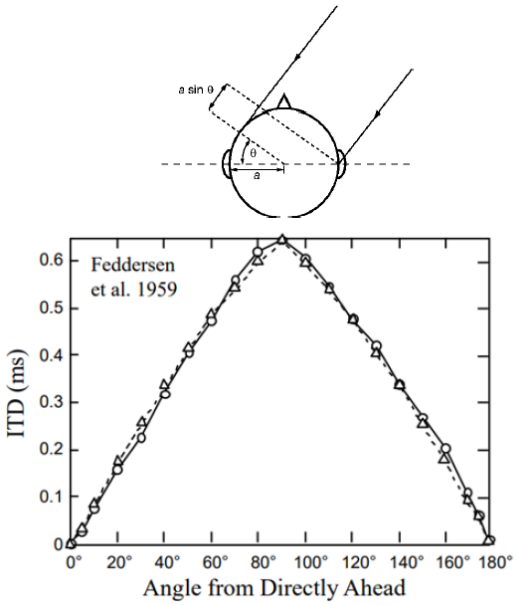


FIGURA 7: ITD

El sistema binaural es sensible a las señales de amplitud de las ILD en cualquier frecuencia, pero son más fiables para frecuencias superiores a 3.000Hz, lo que hace que las señales ILD sean más efectivas a altas frecuencias. Por el contrario, la fisiología binaural utiliza la información de fase de las señales ITD solo a bajas frecuencias, por debajo de 1.500Hz. Para un tono de frecuencia intermedia de 2.000Hz ninguna de las señales funciona bien, es por eso por lo que la capacidad de localización humana es mala para señales que se encuentran en esa región de frecuencias. (Møller 1992)(Park et al. 2004; Physics Today On The Web - Cover Story n.d.)

El conjunto de ITDs e ILDs dependientes de la frecuencia se conocen como la **función de transferencia relacionada con la cabeza** (**HRTF**). La HRTF es una respuesta que caracteriza cómo el oído recibe un sonido desde un punto en el espacio. A medida que el sonido golpea al oyente, el tamaño y la forma de la cabeza, los oídos, el canal auditivo, la densidad de la cabeza, el tamaño y la forma de las cavidades nasales y orales, transforman el sonido y afectan la forma en que se percibe, aumentando algunas frecuencias y atenuando otras.

Se puede usar un par de HRTF para dos oídos para sintetizar un sonido binaural que parece provenir de un punto particular en el espacio. Es una función de transferencia que describe cómo un sonido de un punto específico llegará al oído (generalmente en el extremo del canal auditivo).

Como se ha visto anteriormente los humanos estiman la ubicación de una fuente sonora tomando señales derivadas de un oído (señales monoaurales) y comparando las señales recibidas en ambos oídos (señales binaurales). Las señales monoaurales provienen de la interacción entre la fuente de sonido y la anatomía humana, en la cual la fuente de sonido original se modifica antes de que ingrese al canal auditivo para su procesamiento por el sistema auditivo. Estas modificaciones codifican la ubicación de origen y pueden capturarse mediante una respuesta al impulso que relaciona la ubicación de la fuente sonora y la del oído. Esta respuesta al impulso se denomina **respuesta al impulso relacionada con la cabeza (HRIR)**. La convolución de un sonido aleatorio con la HRIR hace que se escuche el sonido en la posición donde se grabó la HRIR. Las HRIR se han utilizado para producir sonido envolvente virtual. La HRTF es la transformada de Fourier de la HRIR.

Las HRTF para el oído izquierdo y derecho (expresados anteriormente como HRIR) describen el filtrado de una fuente de sonido ($x(t)$) antes de que se perciba en los oídos izquierdo y derecho como $x_L(t)$ y $x_R(t)$, respectivamente.

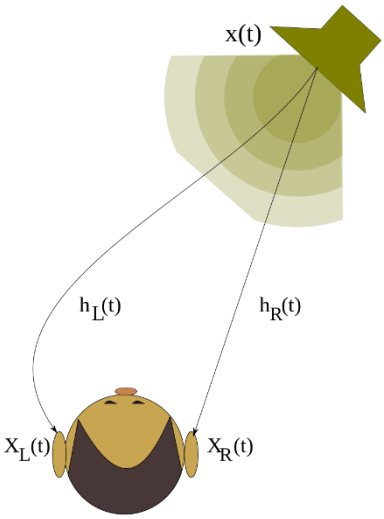


FIGURA 8: HRTF

$$x_L\,(t) = h_L(t) * x(t) \tag{3}$$

$$x_R\,(t) = h_R(t) * x(t) \tag{4}$$

El mecanismo asociado varía entre los individuos, ya que cada individuo tiene unas formas de cabeza y orejas diferente.

La función de transferencia H(f) de cualquier sistema lineal invariante en el tiempo a la frecuencia f es:

$$H(f) = \frac{Salida(f)}{Entrada(f)} \tag{5}$$

Por lo tanto, un método utilizado para obtener la HRTF desde una ubicación de fuente dada es medir la respuesta al impulso relacionada con la cabeza (HRIR), h(t), en el tímpano para el impulso Δ(t) colocado en la fuente.

Las HRTF se miden normalmente en una cámara anecoica para minimizar la influencia de reflexiones tempranas y la reverberación en la respuesta medida. Las HRTF se pueden medir en pequeños incrementos de θ a 1° en el plano horizontal.

Para maximizar la relación señal-ruido (SNR) en un HRTF medido, es importante que el impulso que se genera sea de alto volumen. En la práctica, sin embargo, puede ser difícil generar impulsos a grandes volúmenes y, si se generan, pueden ser dañinos para los oídos humanos, por lo que es más común que las HRTF se calculen directamente en el dominio de la frecuencia utilizando una onda sinusoidal de barrido de frecuencia. o utilizando secuencias de longitud máxima.(Algazi et al. 2001)

Existen varias bases de datos que recogen HRTF para varios sujetos y cabezas binaurales obtenidas en condiciones de laboratorio. Algunas de las más importantes son: SADIE, KEMAR, KU100, CIPIC - U.C. Davies, LISTEN – IRCAM.

KEMAR

KEMAR, el maniquí para pruebas de audífonos e I + D, ha sido diseñado con dimensiones humanas adultas medias y la simulación del oído coincide con la respuesta acústica con una aurícula, un canal auditivo y un tímpano que igualan las dimensiones del oído medio, la impedancia acústica y modos.

Se completó un amplio conjunto de mediciones de la función de transferencia relacionada con la cabeza (HRTF) de un micrófono de cabeza simulada KEMAR. Las mediciones consisten en las respuestas de impulso del oído izquierdo y derecho de un altavoz montado a 1,4 metros del KEMAR.

El receptor, KEMAR, se colocó en siete posiciones diferentes en una sala simulada de dimensiones 6.6 x 8.6 x 3 m. Para estos experimentos, se simula un conjunto de BRIR con los siguientes tiempos de reverberación para cada una de las siete posiciones del receptor.

Las características del receptor del oyente están modeladas con bastante precisión, porque los frentes de ondas acústicas que llegan a las cabezas de los oyentes están representados en los BRIR mediante una respuesta de impulso relacionada con la cabeza adecuadamente retrasada y escalada. Cada respuesta de impulso relacionada con la cabeza se selecciona de una base de datos para el acimut y la elevación. (GRASSound&Vibration n.d.)

FIGURA 9: KEMAR

SADIE

Las HRIR están medidas para cada sujeto en una distribución regular de latitud-longitud (elevación de 15 ° y resolución azimutal variable) y para al menos 12 configuraciones clave de altavoces. Fueron tomadas usando Genelec 8010s en un radio de 1.2m desde el centro de la cabeza del sujeto.

Los datos de audio están disponibles en formato .wav y formato SOFA. Las medidas están etiquetadas por acimut y elevación en grados con precisión de 1 decimal. El acimut se mide en sentido antihorario (a la izquierda) alrededor del plano horizontal. 0° marca directamente en frente del sujeto. La elevación se mide por encima (positivo) y por debajo (negativo) del plano horizontal.

La base de datos SADIE proporciona para 20 sujetos (2 cabezas binaurales y 18 personas):

- Funciones de transferencia relacionadas con la cabeza (formato .wav y SOFA)
- Respuestas de impulso de sala binaural (formato .wav y SOFA)
- Filtros IR / EQ para auriculares (formato .wav)
- Datos antropomórficos
- Una lista de ángulos de medición anecoicos (puntos)
- Gráficos de intensidad, ITD e ILD para HRIR medidos alrededor del plano horizontal

Además de todo esto, SADIE proporciona una selección de archivos de configuración del decodificador Ambisonic, en los cuales se encuentran las matrices de decodificación Ambisonic desde 1er a 5º orden. Cada decodificador es compatible con cada conjunto de datos HRIR. (SADIE | Spatial Audio For Domestic Interactive Entertainment n.d.)

FIGURA 10: SADIE

KU100

El KU100 es un micrófono de cabeza simulada de escucha binaural con auriculares. Las mediciones de HRIR se realizaron la cámara anecoica de la Universidad de Ciencias Aplicadas de Colonia. La cámara tiene unas dimensiones de 4.5 x 11.7 x 2.3m y un límite de frecuencia más bajo de alrededor de 200Hz. La distancia entre el sistema de altavoces y el centro de la cabeza era de aproximadamente 3,25m, lo que puede considerarse como un campo lejano. (NEUMANN n.d.)

FIGURA 11: KU100

CIPIC - U.C. Davies

Está formada por medidas de cuarenta y cinco sujetos diferentes. Posee veinticinco valores de azimuts diferentes y cincuenta elevaciones en incrementos de cinco grados. (HRTF Data – The CIPIC Interface Laboratory Home Page n.d.)

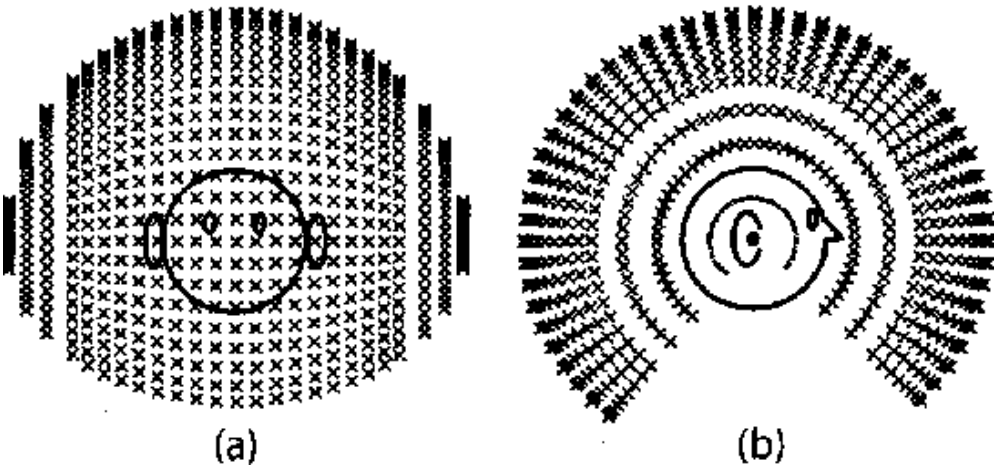


FIGURA 12: CIPIC

LISTEN - IRCAM

Esta base de datos está formada por de medidas tomadas en 51 sujetos diferentes. Por cada sujeto se realizaron 187 medidas, que consisten en 10 ángulos de elevación que van de -45 a +90 grados con incrementos de 15 grados. Los ángulos de acimut varían según el ángulo de elevación desde 24 muestras para una elevación de 0º, hasta una única muestra para elevaciones de 90º. (LISTEN HRTF DATABASE n.d.)

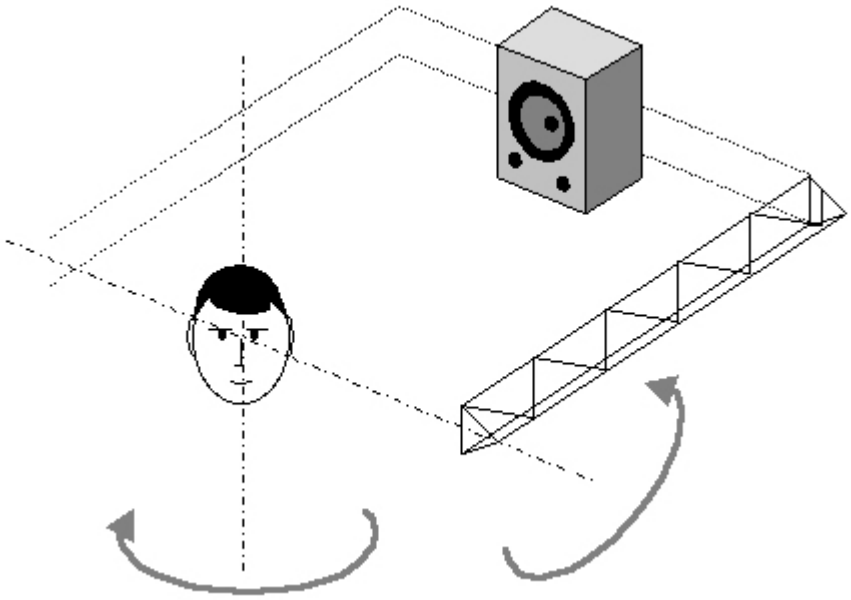


FIGURA 13: LISTEM - IRCAM

La mayoría de estas bases de datos proporciona sus HRTF en formato .wav, aunque algunas, como el caso de SADIE, también permiten descargarlas en formato SOFA.

El formato espacialmente orientado para la acústica ((S)patially (O)riented (F)ormat for (A)coustics: SOFA) Dicho formato de archivo está destinado a leer, guardar y describir datos de sistemas acústicos orientados espacialmente. Ejemplos de datos que consideramos son las funciones de transferencia relacionadas con la cabeza (HRTF), las respuestas de impulso de la sala binaural (BRIR), las mediciones multicanal como las realizadas con conjuntos de micrófonos o los datos de direccionalidad de los altavoces.

Al diseñar SOFA, se establecen los siguientes requisitos:

- Descripción de una configuración de medición con geometría arbitraria, es decir, no limitada a casos especiales como una cuadrícula regular o una distancia constante;
- Los datos autodescriptivos con una definición consistente, es decir, toda la información requerida sobre la configuración de la medición debe

proporcionarse como metadatos en el archivo;
- Flexibilidad para describir datos de múltiples condiciones (oyentes, distancias, etc.) en un solo archivo;
- Archivo parcial y soporte de red;
- Disponible como archivo binario con compresión de datos para almacenamiento y transferencia eficientes;
- Convenciones de descripción predefinidas para las configuraciones de medición más comunes.

Las especificaciones SOFA tienen como objetivo cumplir con todos esos requisitos. En pocas palabras, la configuración de la medición se describe mediante varios objetos y sus relaciones. (Majdak et al. 2013)

También se pueden individualizar las HRTF para cada usuario. Lo más importante para medir el HRTF es un micrófono. En términos de seguridad, precisión y antecedentes teóricos, es apropiado medir el HRTF en la entrada del canal auditivo, utilizando micrófonos tipo tapón auditivo.

FIGURA 14: Micrófono para la individualización de las HRTF

2.2- Virtual surround

El audio de próxima generación **(Next Generation Audio, NGA)** pretende crear una experiencia más inmersiva incluyendo elementos como la elevación e introduce objetos de audio para facilitar el audio inmersivo y personalizado.

Los formatos de **audio basado en canales (Channel Based Audio (CBA))** como estéreo, 5.1... son los más utilizados en la actualidad. En CBA el material de audio se mezcla para un diseño de altavoces específico. Sin embargo, los formatos CBA no cumplen los requisitos establecidos para NGA. Los formatos basados en canales suponen que el sistema de reproducción de sonido del consumidor coincide con el sistema de sonido utilizado para producir el contenido. Esta suposición generalmente no es cierta. Los sistemas NGA deben estar diseñados para reproducir de manera óptima el contenido de audio de acuerdo con el entorno de reproducción del consumidor. Esto incluye, entre otros, la reproducción de auriculares, diseño de altavoces y barras de sonido. CBA no permite esto, de hecho, el contenido CBA es simplemente un conjunto de fuentes de altavoces y no una descripción de la escena de audio. Para abordar estas limitaciones existen dos formatos de audio, **audio basado en objetos (Object Based Audio (OBA)) y audio basado en escenas (Scene Based Audio (SBA)).**

El formato de **audio basado en objetos (Object Based Audio (OBA))** viene en forma de audio sin mezclar componentes (también conocidos como objetos de audio) con metadatos de objetos asociados. Los metadatos del objeto contienen información que caracteriza el objeto de audio, como suposición espacial en la escena y su nivel de audio, según lo previsto por el creador del contenido.

En el sitio de escucha, un renderizador OBA intenta reproducir los objetos de audio basados en la configuración de altavoces específica disponible (que puede no ser estándar). A diferencia de CBA, OBA habilita funciones interactivas al permitir a los usuarios personalizar características específicas de los objetos de audio (si lo permite el creador de contenido y es compatible con el procesador), por ejemplo, alterar la posición espacial o personalizar los niveles de un objeto dado. Para mitigar los problemas de OBA debido a una entrega limitada de ancho de banda, restricciones de complejidad del dispositivo del consumidor y manipulación de escenas, el contenido de audio inmersivo se puede producir y entregar en el SBA.

El formato de **audio basado en escenas (Scene Based Audio (SBA))** se basa en Higher Order Ambisonics (HOA). El contenido en formato SBA está codificado en un conjunto fijo de señales de audio cuyo número es independiente del número de elementos de audio. Al igual que con OBA, el formato SBA es independiente del diseño del altavoz utilizado para la reproducción, permitiendo así la representación de contenido SBA en cualquier diseño de altavoces. Además, el formato SBA permite a los usuarios personalizar e interactuar con el contenido de audio inmersivo. (Olivieri, Peters, and Sen 2019)

Existen diferentes formas de crear un sistema de audio basado en canales. Una de ellas es el virtual surround de escucha por auriculares de escenas sonoras 3D (típicamente 5.1 o 7.1 u otras configuraciones de altavoces diferentes) donde se utilizan las HRTF para la colocación de las fuentes.

Se convoluciona la señal del sonido que se desea colocar por la HRTF de la fuente sonora donde se desea que se reproduzca. El audio de salida binaural será la suma de todos audios que se quieren reproducir convolucionados por sus correspondientes HRTF.

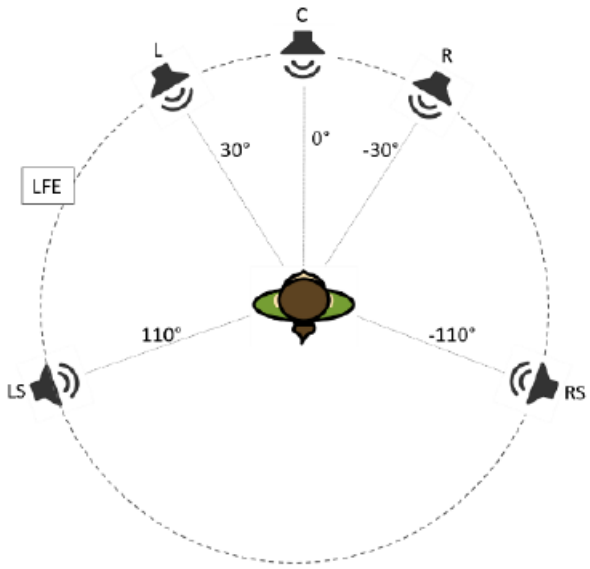


FIGURA 15: Virtual Surround

$$\begin{aligned} y(t) = {} & sL(t) * hL(t) + sC(t) * hC(t) \\ & + sR(t) * hR(t) + sLS(t) \\ & * hLS(t) + sRS(t) * hRS(t) \end{aligned} \tag{6}$$

En un sistema surround de este tipo solo es necesario conocer las HRTF correspondiente a las posiciones de cada altavoz, pero la sensación de cabeza “fija” disminuye la sensación de inmersión. Con la introducción del Head-Tracker se necesita conocer las HRTF de cada orientación para cambiar cada vez las HRTF con las que se convoluciona las señales de entrada en función del ángulo relativo a la nueva orientación de la cabeza.

Si un sonido es reproducido a 30° en la posición del altavoz L, al introducir el Head-Tracker y orientarnos hacia esa dirección, la señal del audio se convolucionará con la HRTF de 0° para que el oyente perciba que el sonido le viene de frente. Esto hace que el sistema sea más inmersivo.

2.3- Ambisonic

La técnica de sonido envolvente de esfera completa es un formato que se conoce como **Ambisonic**. Es una técnica de grabación cuyo objetivo es la síntesis de sonido en tres dimensiones, el cual contempla tanto la grabación y la reproducción como la codificación y decodificación del audio.

El sistema de sonido Ambisonic es una solución tecnológica al problema de codificar direcciones y amplitudes del sonido y reproducirlas con sistemas prácticos de altavoces, de manera que estos simularán las posiciones de las fuentes originales. Esto puede tener lugar a lo largo de un escenario horizontal de 360° o a lo largo de una esfera completa.

La tecnología Ambisonic se divide en 4 fases:

- Grabación con técnicas estereofónicas
- Codificación en formatos específicos
- Decodificación en N canales
- Reproducción a través de m altavoces

A todo ello habría que añadir la posibilidad de rotar la escena Ambisonic con una matriz de rotación y la capacidad de presentar la escena en auriculares en formato mono o binaural. (UNIVERSIDAD POLITÉCNICA DE MADRID ESCUELA UNIVERSITARIA DE INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN PROYECTO FIN DE CARRERA n.d.)

Existen varias técnicas de grabación estereofónicas con sus correspondientes técnicas de codificación y decodificación. Uno de ellos es el formato 2D de primer orden de Ambisonic (FOA). Consiste en una señal correspondiente a un patrón de captación omnidireccional (llamado W), y dos señales correspondientes a los patrones de captación en forma de figura de ocho alineados con los ejes cartesianos X e Y.

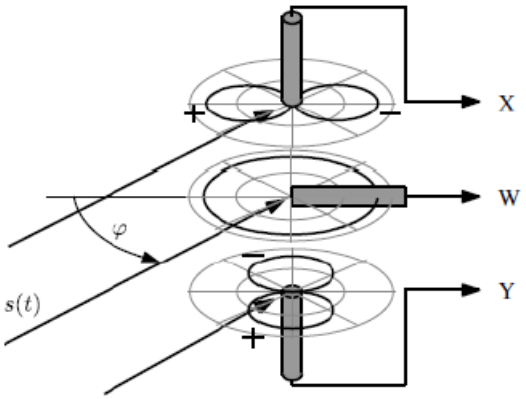


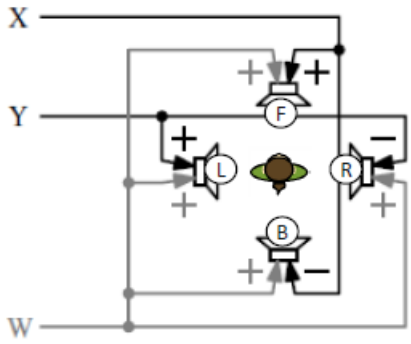
FIGURA 16: Grabación 2D FOA

La codificación se realiza en función del ángulo donde se quiere colocar la fuente y la señal de audio de entrada. El canal W, X e Y de Ambisonics 2D de primer orden se puede reproducir fácilmente en una disposición de cuatro altavoces, frontal, posterior, izquierdo, derecho. Mientras que la contribución de la señal omnidireccional es reproducida por todos los altavoces.

Encoding:

$$\begin{bmatrix} W \\ X \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ cos(\varphi) \\ sen(\varphi) \end{bmatrix} s(t)$$

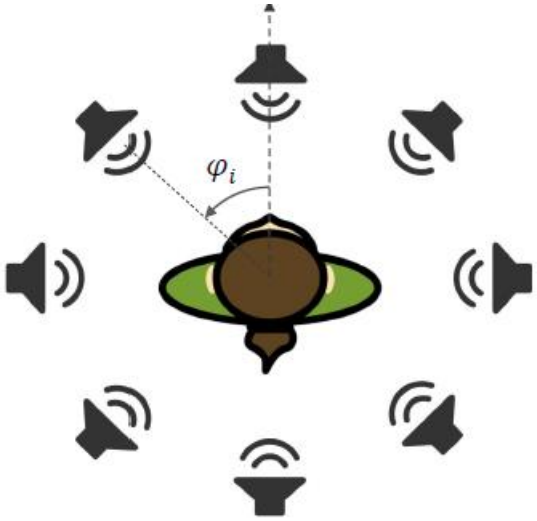
Decoding (4ch):



$$\begin{bmatrix} F \\ L \\ B \\ R \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & -1 & 0 \\ 1 & 0 & -1 \end{bmatrix} \begin{bmatrix} W \\ X \\ Y \end{bmatrix}$$

FIGURA 17: Codificación-Decodificación 2D FOA para 4 canales

Cuando se tiene un numero N de canales la decodificación se realiza de la siguiente manera:



$$\underbrace{\begin{bmatrix} L_1 \\ \vdots \\ L_N \end{bmatrix} = \frac{1}{2}\begin{bmatrix} 1 & cos(\varphi_1) & sen\,(\varphi_1) \\ \vdots & \vdots & \vdots \\ 1 & cos(\varphi_N) & sen\,(\varphi_N) \end{bmatrix}}_{\text{Decoding Matrix}} \begin{bmatrix} W \\ X \\ Y \end{bmatrix}$$

FIGURA 18: Decodificación 2D FOA para N canales

Para rotar la escena de entrada Ambisonics del decodificador, es suficiente obtener un nuevo conjunto de señales 3D forma de figura de ocho mezclando los canales X, Y con la siguiente matriz dependiendo del ángulo de rotación ρ, manteniendo W inalterado. Es aquí donde interviene el Head-Tracker. Con la orientación de la cabeza en el eje Z se determina el ángulo de la matriz de rotación, lo que permite girar la escena en función de la posición del oyente. Dicho ángulo se denominará como “yaw”.

$$\begin{bmatrix} W \\ X' \\ Y' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\rho) & -sen\,(\rho) \\ 0 & sen(\rho) & cos(\rho) \end{bmatrix} \begin{bmatrix} W \\ X \\ Y \end{bmatrix}$$

FIGURA 19: Matriz de rotación de escena sonora en Z (ρ = yaw)

Para construir la matriz de rotación en los tres ejes (X(roll), Y(pitch) y Z(yaw)) hay que combinarlos de la siguiente manera:

$$\mathbf{R}(\phi,\theta,\psi) = \underbrace{\begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{pmatrix}}_{\mathbf{x-axis-rotation(roll)}} \cdot \underbrace{\begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}}_{\mathbf{y-axis-rotation(pitch)}} \cdot \underbrace{\begin{pmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{z-axis-rotation(yaw)}}$$

FIGURA 20: Matriz de rotación de escena sonora en X,Y,Z

Para la representación en 3D de primer orden de Ambisonics (FOA) se añade una señal correspondiente al patrón de captación en forma de figura de ocho alineado con el eje cartesiano Z.

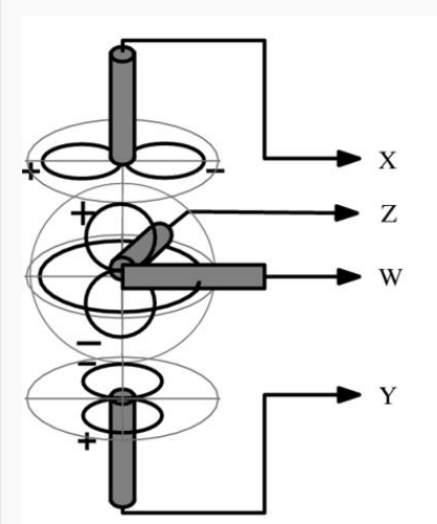


FIGURA 21: Grabación 3D FOA

Para el formato Ambisonic 3D la matriz de codificación no se puede formar en función de un único ángulo (azimut), se necesita también la elevación. Este formato de codificación es conocido como B-format (W, X, Y, Z).

$$\begin{bmatrix} W \\ X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 1 \\ cos(\varphi)cos(\delta) \\ sen(\varphi)cos(\delta) \\ sen(\delta) \end{bmatrix} s(t)$$

FIGURA 22: Codificación 3D FOA

Una vez se tiene la matriz de codificación modificada por la matriz de rotación generada por la posición obtenida por el Head-Tracker se pasa a la decodificación y su posterior presentación en auriculares. En la presentación por auriculares, las señales de los auriculares se generan por convolución con las respuestas de los impulsos relacionados con la cabeza en las posiciones de los altavoces. (Zotter and Frank 2019)

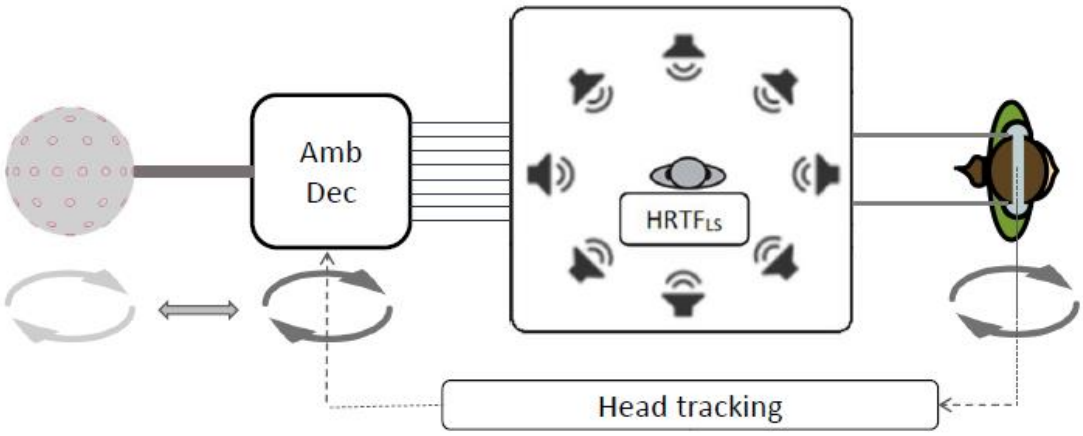


FIGURA 23: Presentación por auriculares

El Head-Tracker también funciona con ordenes mayores de Ambisonic, **Higher Order Ambisonics (HOA)**. HOA se utiliza para reconstruir una onda plana descomponiendo el campo en armónicos esféricos. La codificación en HOA crea un conjunto de señales que dependen de la posición de la fuente de sonido, con los canales ponderados según la dirección de la fuente. Las funciones se vuelven más complejas a medida que aumenta el orden HOA.

Se necesita un número infinito de armónicos esféricos para recrear perfectamente el campo de sonido, pero en la práctica la serie se limita a un orden finito (n). Si n > 1 se considera Ambisonic de orden superior. Se tendrá más precisión contra más armónicos esféricos haya, mayor orden Ambisonic. Además, los armónicos esféricos tienen un grado (m).

Un campo de sonido codificado en HOA requiere $(n + 1)^2$ canales para representar la escena. Ej. 4 para el primer orden, 9 para el segundo, 16 para el tercero, etc. En el caso de la esfera del proyecto es Ambisonic de tercer orden.

La codificación del campo sonoro será una suma de la dependencia espacial correspondiente a los armónicos esféricos y la dependencia temporal de las señales Ambisonic (el peso que se le da a los armónicos). Es necesario asignar pesos distintos a los armónicos esféricos para que la suma de todos ellos cree una esfera unidad.

Para la decodificación se tiene en cuenta la posición de los altavoces, al igual que en Ambisonic de primer orden los diseños regulares en un círculo o esfera son los que producen mejores resultados. El número de altavoces requeridos es al menos el número de canales codificados HOA.

Al igual que Ambisonic de primer orden, es posible hacer rotaciones del campo sonoro completo ya que el número de canales se mantiene independientemente de cuantas fuentes se incluyan.

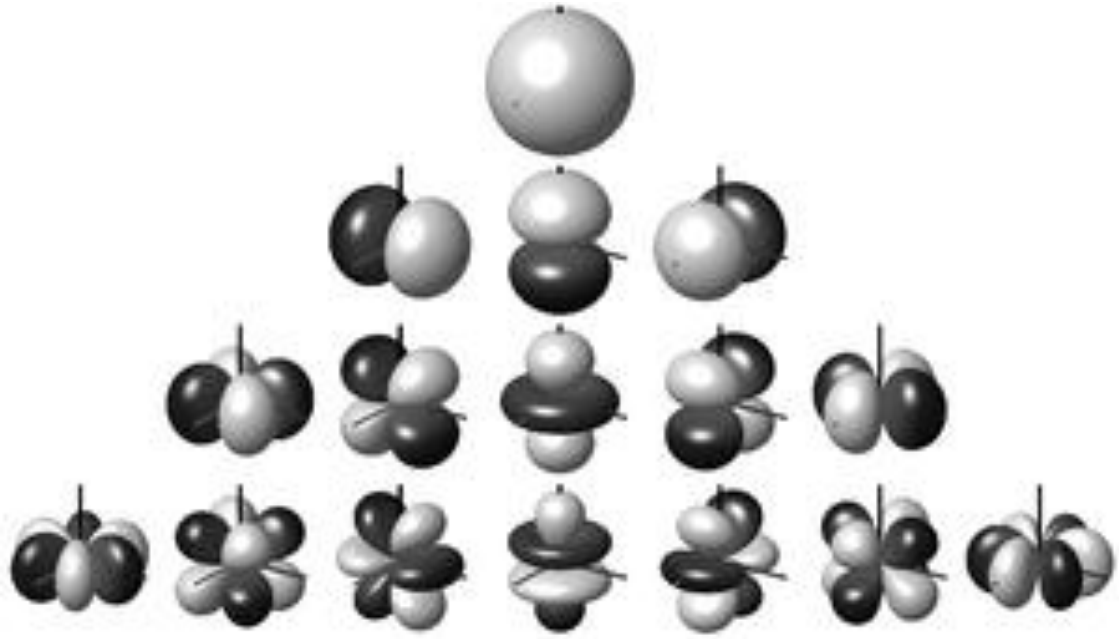


FIGURA 24: High Order Ambisonics (HOA)

3- CULTURA MAKER

3.1- Arduino

Arduino es una plataforma de electrónica open-source basada en hardware y software fácil de usar. Arduino nació principalmente como una herramienta que permitía ser utilizada de manera rápida en prototipos, diseñada para estudiantes sin una base muy profunda en electrónica y programación.

Poco a poco se fue extendiendo gracias a una comunidad muy grande de usuarios y desarrolladores. Fueron apareciendo distintas placas (ver Figura 25) que se adaptaban a las necesidades de los distintos proyectos, diferenciando su oferta de placas simples de 8 bits a productos para aplicaciones IoT. Todas estas placas son de código abierto y cualquiera tiene acceso a los esquemáticos de sus placas a través de la web.

FIGURA 25: Logo Arduino

Los principales motivos por los que se ha elegido Arduino para este proyecto son:

- Es muy económico
- Es multiplataforma
- Tiene un entorno de programación simple y claro
- Software y hardware de código abierto y extensible

En la siguiente tabla se muestran algunas características de los principales modelos de placas que ofrece Arduino.

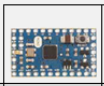
										
Fabricante	Arduino	Arduino	Arduino	Arduino	Arduino	Arduino	Arduino	Netduino	Texas Instruments	Fundación Raspberry Pi
Modelo	Pro Mini	Nano	Uno	Mega / Mega 2560	Leonardo	Micro	Due	Netduino 2	Stellaris Launchpad LM4F120	Raspberry Pi Mod.B
Microcontrolador	AVR Atmega 168 ó 328 8bits	AVR ATmega 168 ó 328 8bits	AVR ATmega 328 8bits	AVR ATmega2560 8bits	AVR ATmega 32u4 8bits	AVR ATmega 32u4 8bits	ARM SAM3X8E Cortex-M3 32bits	ARM STMicro STM32F2 Cortex-M3 32bits	ARM LM4F120H5QR Cortex-M4 32bits	ARM Broadcom BCM2835
Frecuencia	16Mhz	16Mhz	16Mhz	16Mhz	16Mhz	16Mhz	84Mhz	120Mhz	80Mhz	700Mhz
Memoria RAM	2KiB	2KiB	2KiB	8KiB	2.5KiB	2.5KiB	96KiB (64+32KiB)	60KiB	32KiB	512MiB
Memoria EEPROM	1KiB	1KiB	1KiB	4KiB	1KiB	1KiB	0	0	-	-
Memoria FLASH	16 ó 32KiB	16 ó 32KiB	32KiB	128 ó 256KiB	32KiB	32KiB	512KiB	192KiB	256KiB	-
Pines digitales entradas/salidas	14/14	14/14	14/14	54/54	20/20	20/20	54/54	20/20	43/43	8/8
Tensión/corriente pines digitales	3.3v ó 5v 40mA	5v 40mA	5v 40mA	5v 40mA	5v 40mA	5v 40mA	3.3v 3~15mA (130mA entre todos)	3.3v~5v 25mA (125mA entre todos)	5v	-
Pines analógicos entradas/salidas	6/0	8/0	6/0	16/0	12/0	12/0	12/2	6/0	-	-
Tensión/resolución pines analógicos	3.3v ó 5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	5v 10bits (1024 valores)	3.3v 12bits (4096 valores)	5v 12bits (4096 valores)	-	-
Pines con interrupción externa	2	2	2	6	2	2	-	-	-	-
Pines PWM	6	6	6	15	7	7	12	6	-	-
Conexiones Serial / UART	1	1	1	4	1	1	4	4	8	Si
Conexiones I2C / TWI	1	1	1	1	1	1	2	1	4	Si
Conexiones ISP / ICSP	1	1	1	1	1	1	1	1	-	Si
Conexión USB	No (necesita adaptador externo)	Si	Si, USB-B	Si, USB-B	Si, Nativa, MicroUSB	Si, Nativa, MicroUSB	Si, Nativa, MicroUSB	Si, Nativa, MicroUSB	Si, Nativa, MicroUSB	Si, MicroUSB
Conexión USB de depuración	No	No	No	No	No	No	Si, MicroUSB	Si, MicroUSB	Si, MicroUSB	-
Conexión Bluetooth	No	No	No	No	No	No	No	No	No	-
Conexión WiFi	No	No	No	No	No	No	No	No	No	-
Conexión Ethernet	No	No	No	No	No	No	No	No	No	Si
Conexión USB Host	No	No	No	No	No	No	Si	No	Si	Si
Almacenamiento por SD	No	No	No	No	No	No	No	No	No	Si
Corriente en el pin de 5v	-	500mA	500~800mA	500~800mA	500~800 mA	500mA	800mA	-	-	-
Corriente en el pin de 3.3v	-	50mA	50mA	50mA	50mA	50mA	800mA	-	-	-
Voltaje de alimentación por el USB	3.3v ó 5v (sin usb)	5v	5v	5v	5v	5v	5v	5v	5v	5v
Voltaje de alimentación recomendado por el Jack	3.35 -12 V (modelo 3.3V) ó 5 - 12 V (modelo 5V)	7~12v	7~12v	7~12v	7~12v	7~12v	7~12v	7.5~9v	-	-
Voltaje de alimentación limite por el Jack	-	6~20v	6~20v	6~20v	6~20v	6~20v	6~20v	-	-	-
Precio oficial	15+gi	-	20€+gi	40€+gi	18€+gi	18€+gi	39€+gi	~35$+gi	13$+gi	~43$+gi
Precio BBB	~4€	~9€	~10€	~12€	11€~	~16€	~38€	25~30€	~15€	~35€

FIGURA 26: Características de las principales placas Arduino

Para este proyecto se van a utilizar dos placas Arduino. Una de ellas es la placa Arduino UNO, es el modelo de referencia de Arduino.

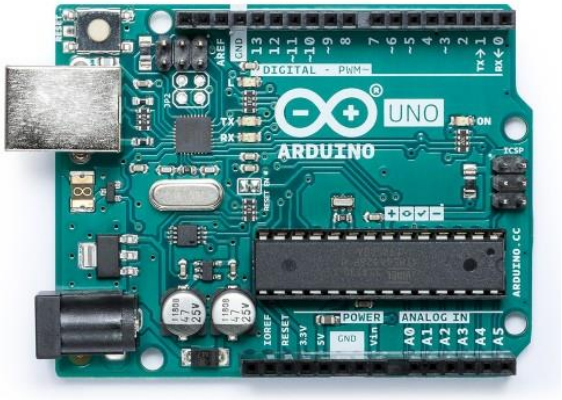


FIGURA 27: Arduino UNO

Dispone de un procesador ATMega328, al igual que la mayoría de las placas de Arduino. Tiene 14 pines de entrada / salida digital (de los cuales 6 se pueden usar como salidas PWM), 6 entradas analógicas, un resonador cerámico de 16MHz, una conexión USB, un conector de alimentación, un encabezado ICSP y un botón de reinicio.

Para alimentarlo se puede conectar a un ordenador por el cable USB, a través de un adaptador de CA a CC o de una batería.

Para una de las partes del Head-Tracker era necesario utilizar una placa con menor tamaño y que redujera el consumo manteniendo las mismas prestaciones. Para ello se utilizó un Arduino NANO.

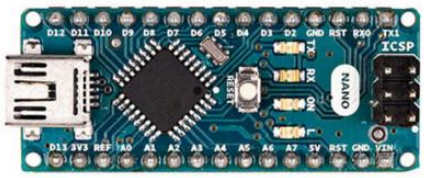


FIGURA 28: Arduino NANO

Dispone de un procesador ATMega328, al igual que la paca Arduino UNO. Además, dispone de un conector de alimentación de CC y funciona con un cable USB Mini-B.

El software de código abierto de Arduino proporciona una forma fácil y rápida de escribir código y subirlo a las placas. Es compatible con cualquier sistema Windows, Mac o Linux y funciona para cualquier placa Arduino del mercado.

El Arduino IDE (integrated development environment) es un programa informático compuesto por un conjunto de herramientas de programación. Puede dedicarse en exclusiva a un solo lenguaje de programación o bien puede utilizarse para varios..

Un IDE es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, que consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Además, en el caso de Arduino, incorpora las herramientas para cargar el programa ya compilado en la memoria flash del hardware a través del puerto serie.

Los programas de Arduino están compuestos por un solo fichero con extensión “ino”, aunque es posible organizarlo en varios ficheros. El fichero principal siempre debe estar en una carpeta con el mismo nombre que el fichero. Cuenta con un gestor de actualizaciones y librerías que permiten mantenerlo actualizado. Dispone de diferentes programas de ejemplo para empezar a usar Arduino.

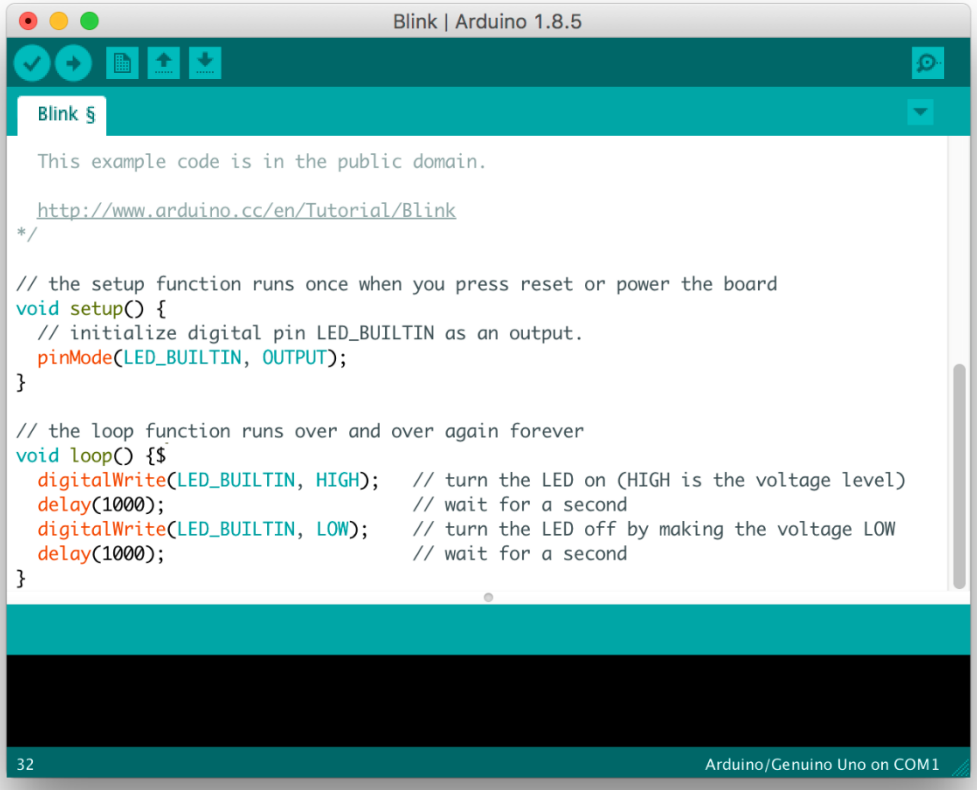


FIGURA 29: Arduino IDE

Para poder ver en el ordenador los resultados que se van obteniendo, Arduino ofrece un monitor serie que escribe por pantalla lo que se le pasa en el código y un serial plotter que muestra los resultados en forma de gráfica. (Arduino - Home n.d.)

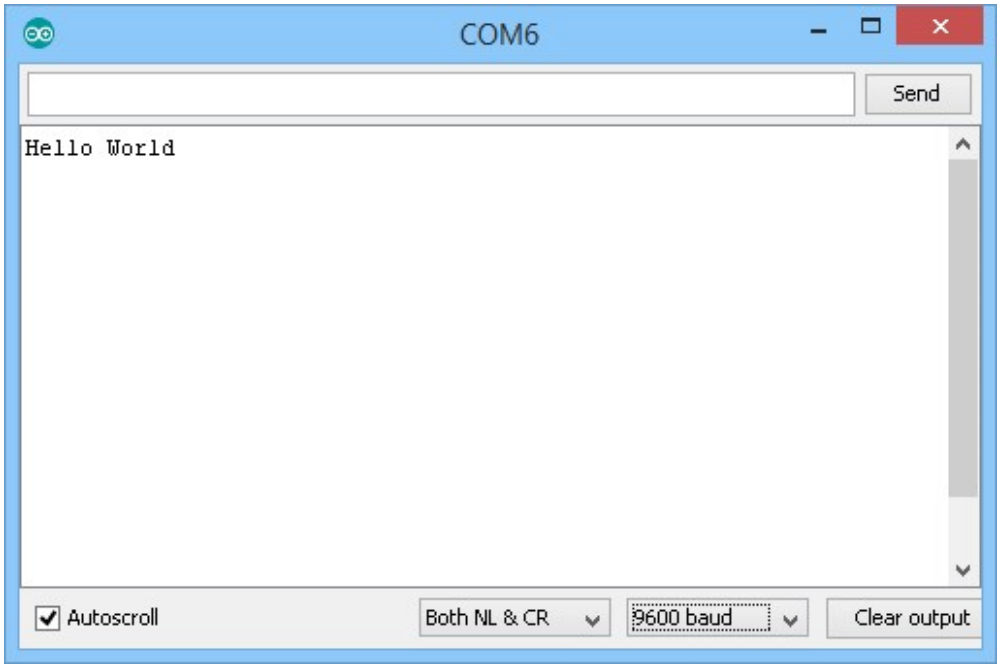


FIGURA 30: Monitor Serie Arduino

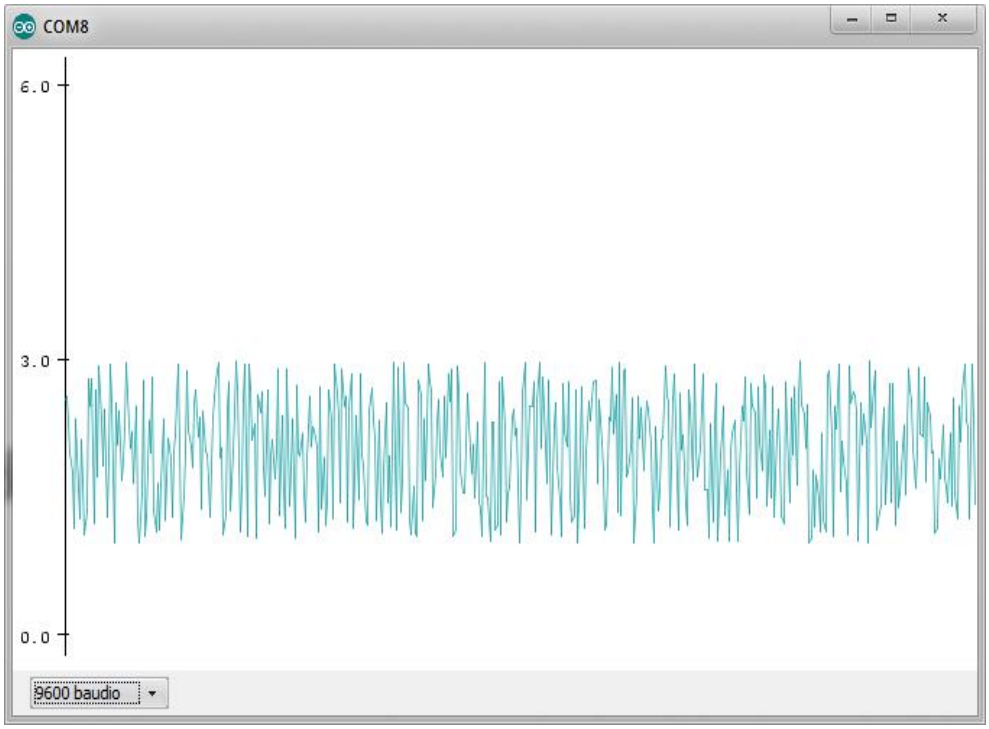


FIGURA 31: Serial Plotter Arduino

3.2- Impresión 3D

La impresión 3D es un conjunto de tecnologías de fabricación aditiva. La fabricación aditiva es un proceso mediante el cual se va añadiendo material para fabricar un objeto tridimensional. La escultura sería el contrario, se parte de un bloque y se le quita material hasta conseguir el objeto deseado.

Existen diferentes estilos de Impresión 3D.

Impresión 3D FDM

Funciona con un rollo de filamento de plástico, el cual se funde en una boquilla muy caliente y sale un hilo muy pequeño. Ese hilo se va colocando con una forma determinada hasta obtener el objeto que se ha diseñado. Sus principales ventajas son:

- Tecnología muy accesible
- Bastante económica
- Mantenimiento y uso de las máquinas muy bajo

Impresión 3D SLA

No se usa una bobina de filamento de plástico sino una resina líquida. Esta resina se inserta en una cubeta y un láser (un dispositivo con luz ultravioleta) solidifica esa resina creando el objeto tridimensional. Sus ventajas son:

- Calidad muy alta, pero proceso más químico
- Se pueden crear piezas más pequeñas

Impresión 3D SLS

Utiliza el mismo proceso que la impresión 3D SLA, pero en este caso la resina líquida se sustituye por un polvo de nylon. Se obtienen muy buenos resultados, pero es una técnica muy cara.

Analizando las características de las diferentes técnicas de impresión 3D, la impresión 3D FDM será la que más se ajusta a este proyecto.

Existen una gran variedad de materiales plásticos que se pueden utilizar para la impresión 3D FDM, pero de entro todos ellos cabe destacar el ácido poliláctico (PLA). El PLA es un poliéster biodegradable y bioactivo compuesto por componentes básicos de ácido láctico. Sus principales ventajas para la impresión 3D son:

- Es el más sencillo

- Se imprime con gran facilidad
- Necesita una temperatura muy baja
- Se puede postprocesar y pintar muy fácil

Una impresora FDM consta de diferentes partes esenciales que en su conjunto hacen que la impresión se lleve a cabo con precisión.

La estructura suele ser de metal, es la parte que le da rigidez a la impresora. Cuanto más rígida mayor es la calidad que tiene la pieza final.

El movimiento se realiza en los tres ejes de coordenadas cartesianas.
Eje X. Realiza el movimiento izquierda – derecha y suele llevar el cabezal.
Eje Y. Realiza el movimiento delante – atrás y suele llevar la base donde se imprime
Eje Z. Realiza el movimiento abajo – arriba suele coincidir con el eje X.

Si la impresora es de filamento habrá una bobina que puede ser de diferentes materiales, que se introduce dentro de la impresora con un motor. Este motor tiene una rueda que empuja el filamento por un tubo hasta el sistema de extrusión. En él se funde el filamento con una resistencia a unos 200° y sale por la parte inferior.

Por último, la impresora 3D se comunica con el ordenador mediante una tarjeta microSD. En ella se meten los archivos con el diseño tridimensional que se desea construir. En la mayoría de las impresoras, existe una pantalla controlada por un botón donde se pueden controlar todos los procesos (iniciar la impresión, pausarla, cambiar el material, calentar o enfriar el extrusor).

FIGURA 32: Impresora 3D FDM

Existen diferentes herramientas para el diseño 3D y modelado. Entre ellas cabe destacas **Autodesk Fusion 360**, es una herramienta integral de diseño de productos.

Fuision 360 tiene la opción de simular para comprobar si algo funciona antes de imprimirlo. El diseño generativo de Fusion 360 permite, a partir de un diseño normal, determinar cómo reducir la cantidad de material manteniendo las propiedades.

El diseño generado se guarda en la nube, donde puede compartirse para que puedan modificarlo otros usuarios. También permite sacar planos del diseño.

Fusion 360 posee un entorno de trabajo que permite generar cualquier diseño utilizando las diversas formas y modificándolas para obtener el objeto deseado.

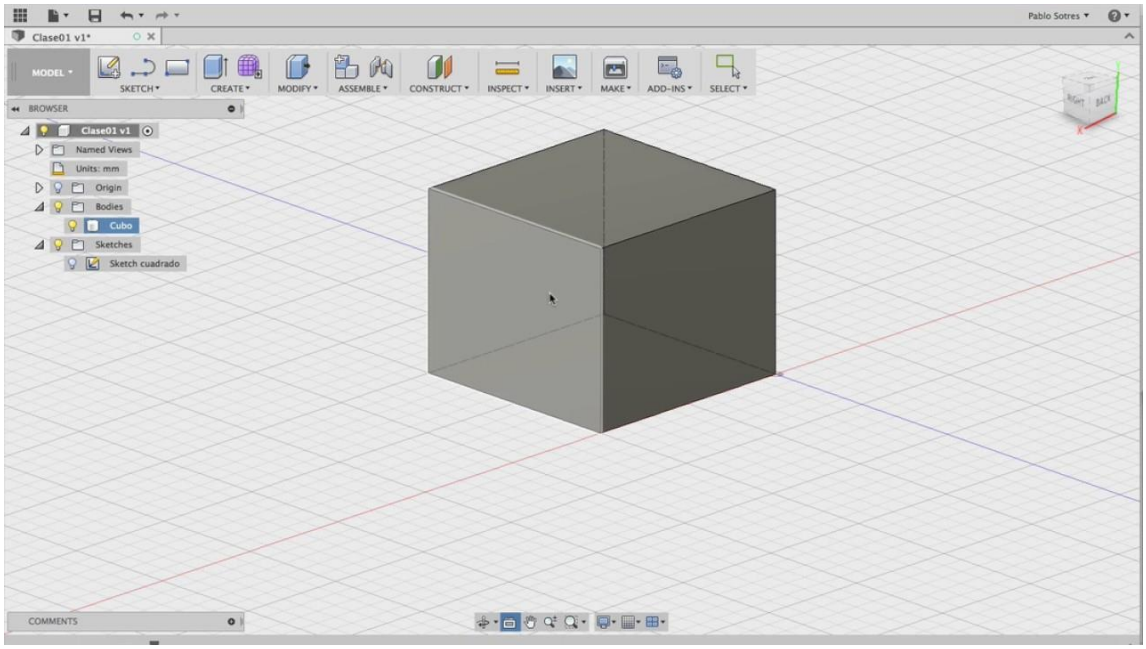


FIGURA 33: Autodesk Fusion 360

Una vez creado el modelo 3D hay que prepararlo para que la impresora 3D pueda fabricarlos. De esta traducción se encarga Ultimaker Cura.

Cura es un software para procesar los archivos de diseño en 3D y generar cada capa que será fabricada por la impresora 3D, de forma que se obtiene un archivo que será transmitido a la impresora 3D.

Cura es de código abierto y ofrece la posibilidad de configurarlo para adaptarse a cada impresora. Dispone de opciones avanzadas para el control del relleno y las velocidades de movimiento, impresión y retracción. (Introducción al diseño e impresión en 3D (Agustín “Flowalistik” Arroyo). Curso online | Domestika n.d.)

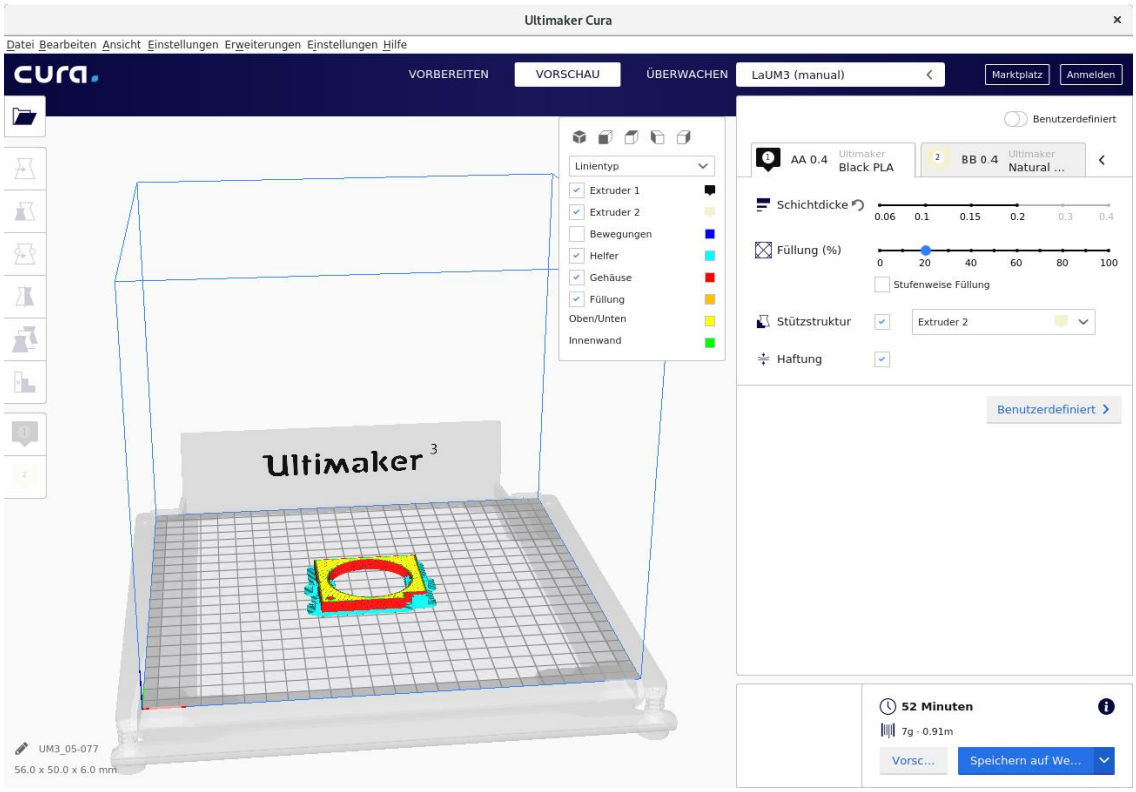


FIGURA 34: Ultimaker Cura

4- DISEÑO Y CONSTRUCCIÓN DEL DISPOSITIVO

4.1- Estado del arte

En el mercado existen diferentes dispositivos Head-Tracker ideados por diferentes desarrolladores de una forma maker. Dentro del mundo empresarial, cabe destacar Waves Audio, es un desarrollador y proveedor de tecnologías profesionales de procesamiento de señal de audio digital y efectos de audio. Wave audio dispone de más de 200 productos de software dedicados a la producción musical, ingeniería, mezcla y masterización.

Waves Audio ha desarrollado el dispositivo de Head-Tracker Nx. Es una tecnología de software que aplica imágenes panorámicas de audio a auriculares estéreo, lo que abre la puerta a aplicaciones de realidad virtual, realidad aumentada y mezcla 3D. Waves Nx sirve de puente recreando en los auriculares las mismas señales auditivas que llegan a los oídos en el mundo real. Una de las maneras en que lo hace es siguiendo la posición de la cabeza del usuario y ubicando el sonido en los auriculares del usuario para igualar el modo en que el usuario lo oiría en el mundo real. Esta tecnología sigue los movimientos reales de la cabeza del usuario, reproduciendo los mínimos matices, lo que crea una experiencia sensorial real, dinámica y variable, posicionando de manera apropiada el audio en el auricular izquierdo y derecho para simular movimiento a través de un espacio de sonido tridimensional.

FIGURA 35: Waves Nx

Waves Nx dispone de un plugin de monitoreo virtual que replica un espacio acústico en los auriculares del oyente. Permite escuchar el audio con precisión en el espacio tal cual se escucha en el punto central de una sala. Tiene la posibilidad de representar escenas sonoras típicas como 5.1. Si se le añade una cámara al sistema se puede medir la posición en relación con la sala virtual en tiempo real. Con todo ello se analiza con precisión los niveles, las frecuencias, la profundidad de la escena sonora. (Nx – Virtual Mix Room over Headphones | Waves n.d.)

FIGURA 36: Waves Nx plugin

4.2- Acelerómetro MPU6050

La aceleración es la variación de la velocidad por unidad de tiempo es decir razón de cambio en la velocidad respecto al tiempo. Así mismo, la segunda ley de Newton indica que un cuerpo con masa constante, la aceleración de dicho cuerpo es proporcional a la fuerza que actúa sobre él mismo. Este segundo concepto es utilizado por los acelerómetros para medir la aceleración.

El módulo MPU6050 contiene un giroscopio de tres ejes con el que se puede medir la velocidad angular y un acelerómetro de 3 ejes con el que medimos los componentes X, Y y Z de la aceleración.

El MPU6050 es un sensor de movimiento que tiene un conversor ADC de 16 bits que convierte los datos a un valor digital, el módulo del giroscopio se comunica con el Arduino a través de la comunicación serie I2C a través del reloj serial (SCL) y datos (SDA). El chip MPU6050 necesita 3,3V pero un regulador de la tarjeta permite alimentarlo a 5V.

El procesador interno del IMU (Inertial Measurment Units) es capaz de realizar cálculos precisos de los valores que miden sus sensores internos que son, aceleraciones lineales y angulares, para informarnos de valores útiles como los ángulos de inclinación con respecto a los 3 ejes principales.

La dirección de los ejes está indicada en el módulo el cual hay que tener en cuenta para no equivocarse en el signo de las aceleraciones.

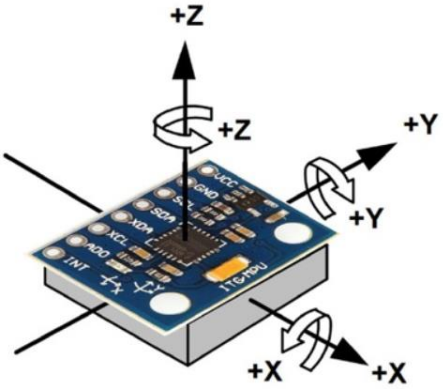


FIGURA 37: Ejes MPU6050

El módulo MPU6050 estará conectado a la placa NANO de Arduino de la siguiente manera:

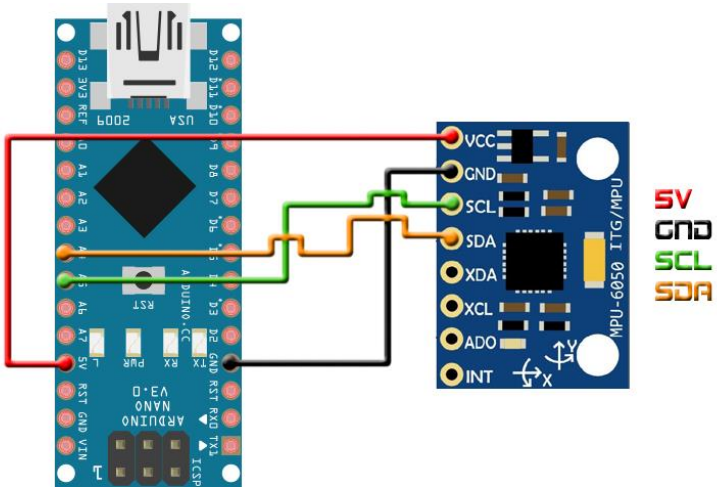


FIGURA 38: Conexión Arduino NANO y MPU6050

El módulo MPU6050 con la placa NANO estarán situados en el dispositivo que llevará el oyente en la cabeza, leerá el valor de estos tres ejes para determinar la posición de la cabeza y el movimiento.

La lectura de estos tres ejes X, Y y Z se recibe en el puerto serie. Si no se procesa son valores que sirven de muy poco. Para ello se hace una conversión a otro sistema de coordenadas.

Se utiliza un sistema de ángulos de navegación, son un tipo de ángulos de Euler para

describir la orientación de un objeto en tres dimensiones. Los tres ángulos que los componen son la dirección (yaw), elevación (pitch) y el ángulo de alabeo (roll). Existen tres rotaciones principales, normalmente llamadas igual que el eje sobre el que se producen.

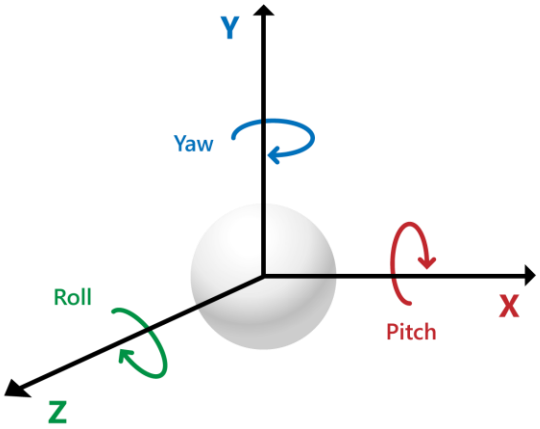


FIGURA 39: Yaw, Pitch y Roll

Es importante calibrar el módulo MPU6050 cuando se encuentre en la posición inicial para que la identifique como la (0, 0, 0). Esto es necesario ya que el sensor MPU6050 probablemente no se encuentre en una posición horizontal, debido a que el sensor al ser soldado en el módulo puede estar desnivelado. Puede pasar lo mismo cuando se coloca el módulo en el Head-Tracker. Con la calibración se solucionan todos estos problemas.

Una vez se tiene la posición del oyente en este sistema de coordenadas ya se puede implementar con otras aplicaciones para poder interactuar con ellas. (Tutorial MPU6050, Acelerómetro y Giroscopio n.d.)

4.3- Bluetooth HC-05

Uno de los objetivos que se fijaron en este proyecto era que el dispositivo Head-Tracker funcionará de forma inalámbrica. Para ello se escogió la tecnología Bluetooth. Se utilizan módulos HC-05, son módulos Bluetooth para trasmitir datos entre dispositivos y que solucionan el problema de la comunicación inalámbrica con Arduino.

La comunicación Bluetooth es similar al uso del puerto serie normal. La diferencia es que, en lugar de usar un cable, hay que emparejar el módulo con el dispositivo. Para establecer la comunicación desde el dispositivo, se puede usar el propio Serial Monitor de Arduino IDE.

El módulo HC-05 estará conectado a la placa NANO de Arduino de la siguiente manera:

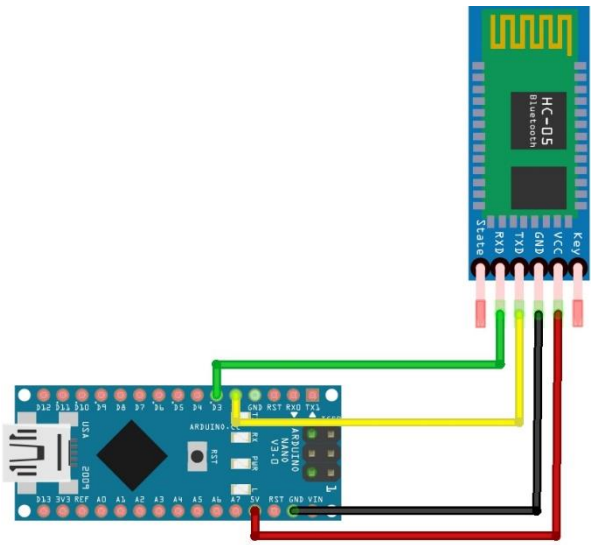


FIGURA 40: Conexión Arduino NANO y HC-05

El otro dispositivo HC-05 que recibirá la información, estará conectado a la placa Arduino UNO que tiene entrada por el puerto Serie al PC de la siguiente manera:

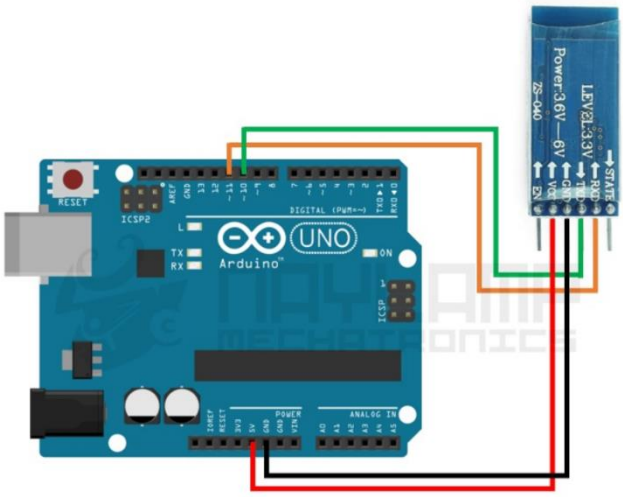


FIGURA 41: Conexión Arduino UNO y HC-05

La comunicación Bluetooth se da entre dos tipos de dispositivos: un maestro y un esclavo.

Cuando un dispositivo está configurado como esclavo espera que un dispositivo Bluetooth maestro se conecte a este, generalmente se utiliza cuando se necesita comunicarse con un Pc.

Si está configurado como maestro, este HC-05 es el que inicia la conexión. Un dispositivo maestro solo se puede conectar a un dispositivo esclavo. Generalmente se utiliza este modo para comunicarse entre módulos Bluetooth. Pero es necesario especificar antes con que dispositivo se tiene que conectar.

El módulo HC-05 viene con una configuración por defecto que se puede cambiar.

Existen 4 estados diferentes para el módulo HC-05:

Estado desconectado: entra en este estado cuando alimentas el módulo y no se ha establecido ninguna conexión. El LED parpadea rápidamente y no se pueden interpretar los comandos AT.

Estado conectado: cuando se establece una conexión con otro dispositivo. El LED hace un doble parpadeo. Todos los datos que se ingresen al HC-05 por el Pin RX se trasmiten por bluetooth al dispositivo conectado, y los datos recibidos se devuelven por el pin TX.

Modo AT 1: para entrar en este estado después de conectar y alimentar el módulo es necesario presionar el botón del HC-05. Permite enviar comando AT a la velocidad que está configurado. El LED parpadea rápidamente.

Modo AT 2: para entrar en este estado es necesario tener presionado el botón al momento de alimentar el módulo. Para enviar comandos AT es necesario hacerlo a la velocidad de 38400 baudios. El LED parpadea rápidamente.

Para configurar los dispositivos HC-05 para este proyecto se utiliza el Modo AT 2. Se abre el Monitor serial del IDE de Arduino. Para poder leer las respuestas hay que escoger “Ambos NL & CR” y la velocidad “38400 baudios” (la velocidad para comunicarse en el Modo AT 2). Una vez hecho esto se pueden empezar a enviar los comandos AT al dispositivo HC-05. Los distintos comandos de configuración son:

```
*****************************************************************************
* Command              Description
*---------------------------------------------------------------------------
* AT                   Check if the command terminal work normally
* AT+DEFAULT           Restore factory default
* AT+BAUD              Get/Set baud rate
* AT+RESET             Software reboot
* AT+ROLE              Get/Set current role.
* AT+DISC              Disconnect connection
* AT+ADVEN             Broadcast switch
* AT+ADVI              Broadcast interval
* AT+NINTERVAL         Connection interval
* AT+POWE              Get/Set RF transmit power
* AT+NAME              Get/Set local device name
* AT+LADDR             Get local bluetooth address
* AT+VERSION           Get firmware, bluetooth, HCI and LMP version
* AT+TYPE              Binding and pairing settings
* AT+PIN               Get/Set pin code for pairing
* AT+UUID              Get/Set system SERVER_UUID .
* AT+CHAR              Get/Set system CHAR_UUID .
* AT+INQ               Search from device
* AT+RSLV              Read the scan list MAC address
* AT+CONN              Connected scan list device
* AT+CONA              Connection specified MAC
* AT+BAND              Binding from device
* AT+CLRBAND           Cancel binding
* AT+GETDCN            Number of scanned list devices
* AT+SLEEP             Sleep mode
* AT+HELP              List all the commands
* ---------------------------------------------------------------------------
*****************************************************************************
```

FIGURA 42: Comandos AT del dispositivo HC-05

El dispositivo HC-05 conectado a la placa Arduino NANO que envía la información del acelerómetro es el Maestro y se ha configurado como:

- Modo o role: Maestro
- Nombre: Head
- Código de emparejamiento: 1212
- Velocidad: 38400 baudios

El dispositivo HC- 05 conectado a la placa Arduino UNO que recibe la información y se la pasa al Pc es el Esclavo y se ha configurado como:

- Modo o role: Esclavo
- Nombre: Pc

- Código de emparejamiento: 1212 (la misma que el otro dispositivo)
- Velocidad: 38400 baudios

Una vez se han configurado ambos dispositivos HC-05 ya se puede realizar la comunicación Bluetooth entre ellos para enviar datos. (Configuración del módulo bluetooth HC-05 usando comandos AT n.d.)

4.4- Comunicación OSC plugins Reaper

Reaper es una estación de trabajo de audio digital y un software secuenciador MIDI. Actúa como un host para la mayoría de los formatos de plugin (VST, IEM...) En este proyecto se utiliza Reaper porque permite visualizar todos los resultados obtenidos por el Head-Tracker.

En Reaper se crearán 3 pistas. Headphones (salida por auriculares), Loudspeakers (salida por altavoces) y Ambisonics (escena Ambisonic). En la pista Ambisonic se configurarán el número de canales según el orden de Ambisonic que se desee. Habrá que enrutar el bus de Ambisonic tanto a Loudspeakers como a Headphones seleccionando la fuente multicanal con los canales de Ambisonic que se hayan elegido para que todos los canales del bus estén ruteados. Este enrutamiento sirve para decodificar la escena Ambisonic tanto a altavoces como a auriculares.

Para poder empezar a realizar pruebas hay que cargar pistas de audio y rutearlas al bus Ambisonic. Una vez hecho este se pasa a cargar los plugins deseados en cada bus. En el bus Ambisonic se pondrá el plugin con el que se quiere interactuar. El bus de Loudspeakers no será de interés para esta aplicación ya que reproduciremos por auriculares. En el bus Headphones se encontrará el plugin 'Binaural Decoder' de IEM para la decodificación de la señal Ambisonic.

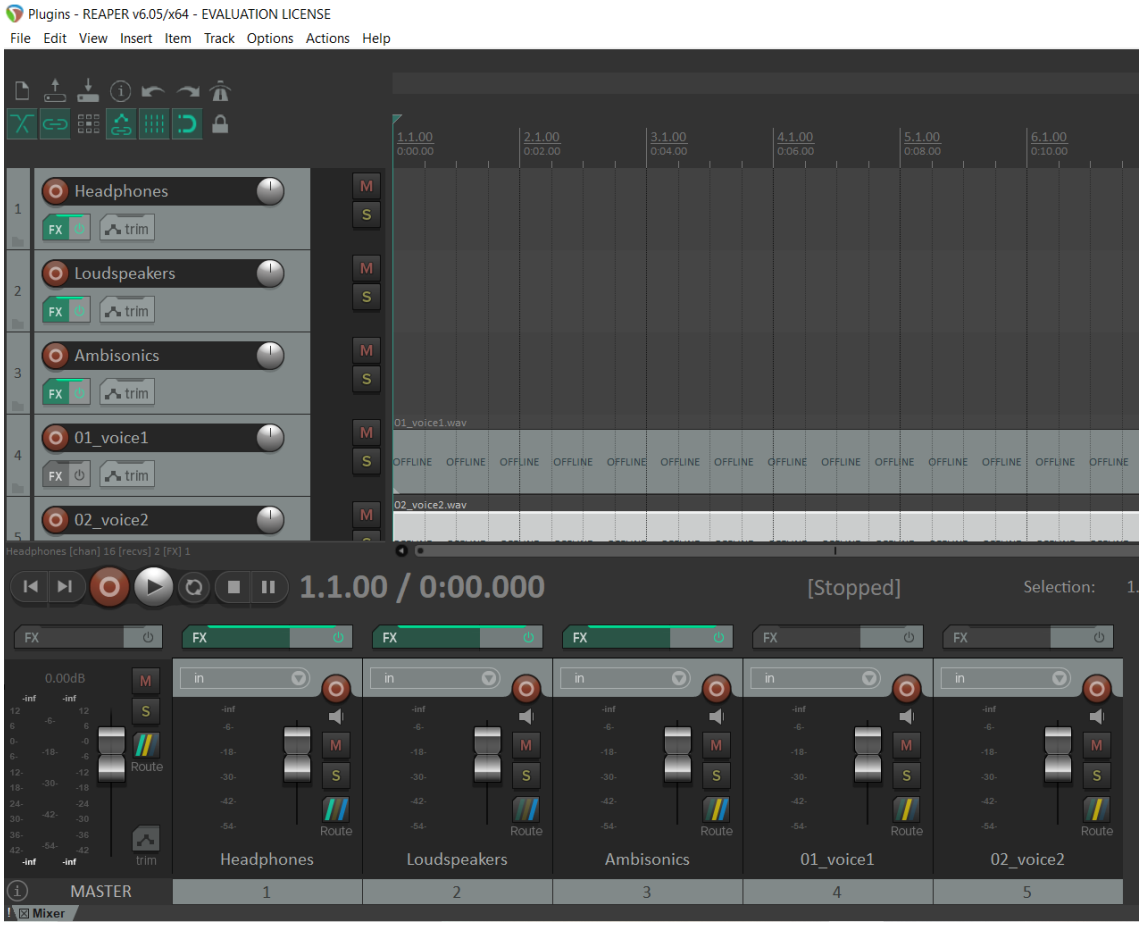


FIGURA 43: Entorno de trabajo de Reaper

Para que los plugins de Reaper puedan comunicarse con el Head-Tracker existen los Open Sound Control (OSC). El OSC es un protocolo para comunicación entre ordenadores, sintetizadores musicales y otros dispositivos multimedia. Sus principales ventajas son, independencia del medio de transmisión, transportar cualquier tipo de datos y permite control en tiempo real de sonido y su procesado.

Para poder recibir e interpretar estos mensajes es necesario generar una aplicación de servidor OSC, ésta escucha mensajes OSC, a través de su IP y un número de puerto. Para enviar mensajes es necesario generar un cliente OSC, éste se conecta a la IP y el puerto correspondiente a un servidor OSC.

El procedimiento habitual es iniciar el servidor OSC, pedir la conexión, enviar datos desde el cliente y cerrar la conexión cuando sea pertinente. El servidor descifra las rutas y agrupa los datos a medida que entran los mensajes. Para recibir y enviar mensajes hay que tener un servidor y un cliente trabajando a la vez.

Para realizar la comunicación entre el dispositivo Head-Tracker y Reaper se ha utilizado

un Bridge que recibe la posición del oyente como roll-pitch-yaw y se la envía al plugin que se esté utilizando en Reaper.

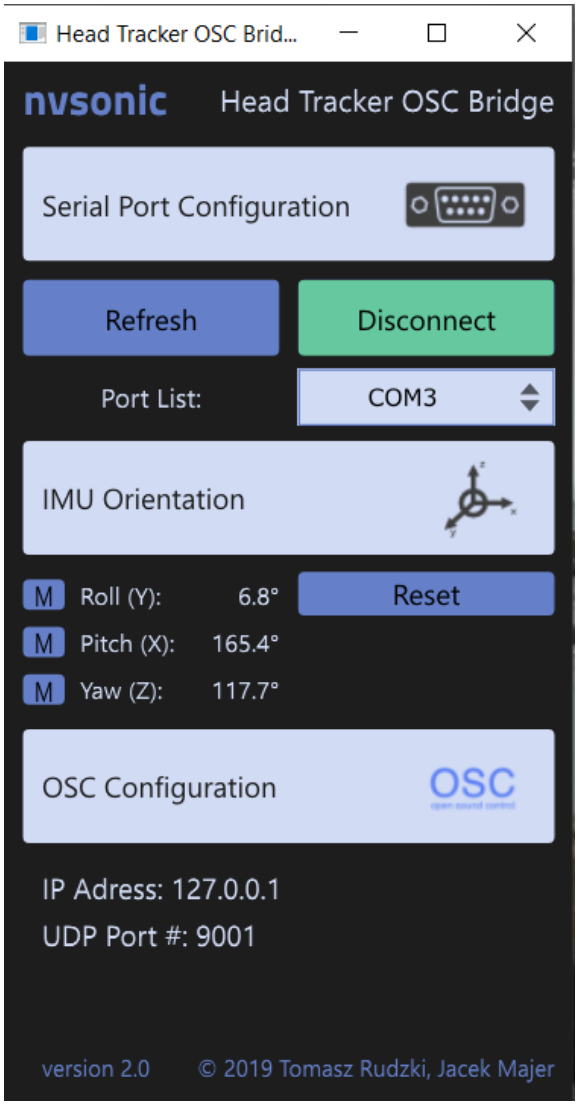


FIGURA 44: Bridge Arduino-Reaper

Para que Reaper reciba los datos del Bridge hay que seguir los siguientes pasos: Options -> preferencias -> control/osc -> add-> OSC -> Logical Port (recibe only) -> Puerto 9001 (el del Bridge) -> Activar la opción “Aflow brinding”.

A partir de aquí Reaper ya recibirá la lectura del Head-Tracker. Ahora hay que interactúe con el plugin. Como ejemplo el Scene Rotator de IEM. Los pasos que hay que seguir para asociar la entrada OSC al plugin son: Param -> Fx parameter -> Learn -> seleccionamos el ángulo que se desea aprender y se silencian los demás en el Bridge.

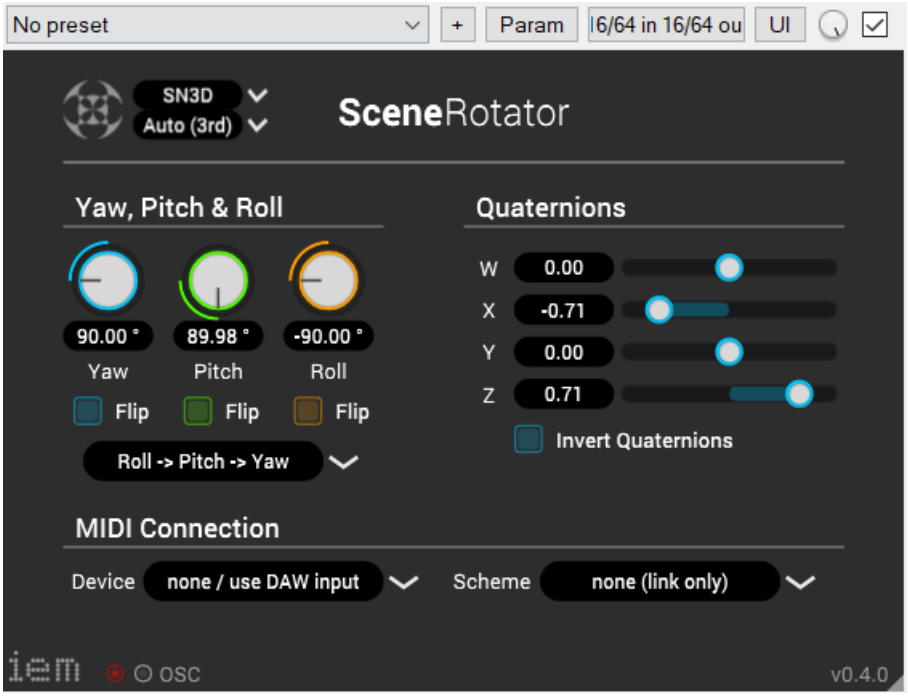
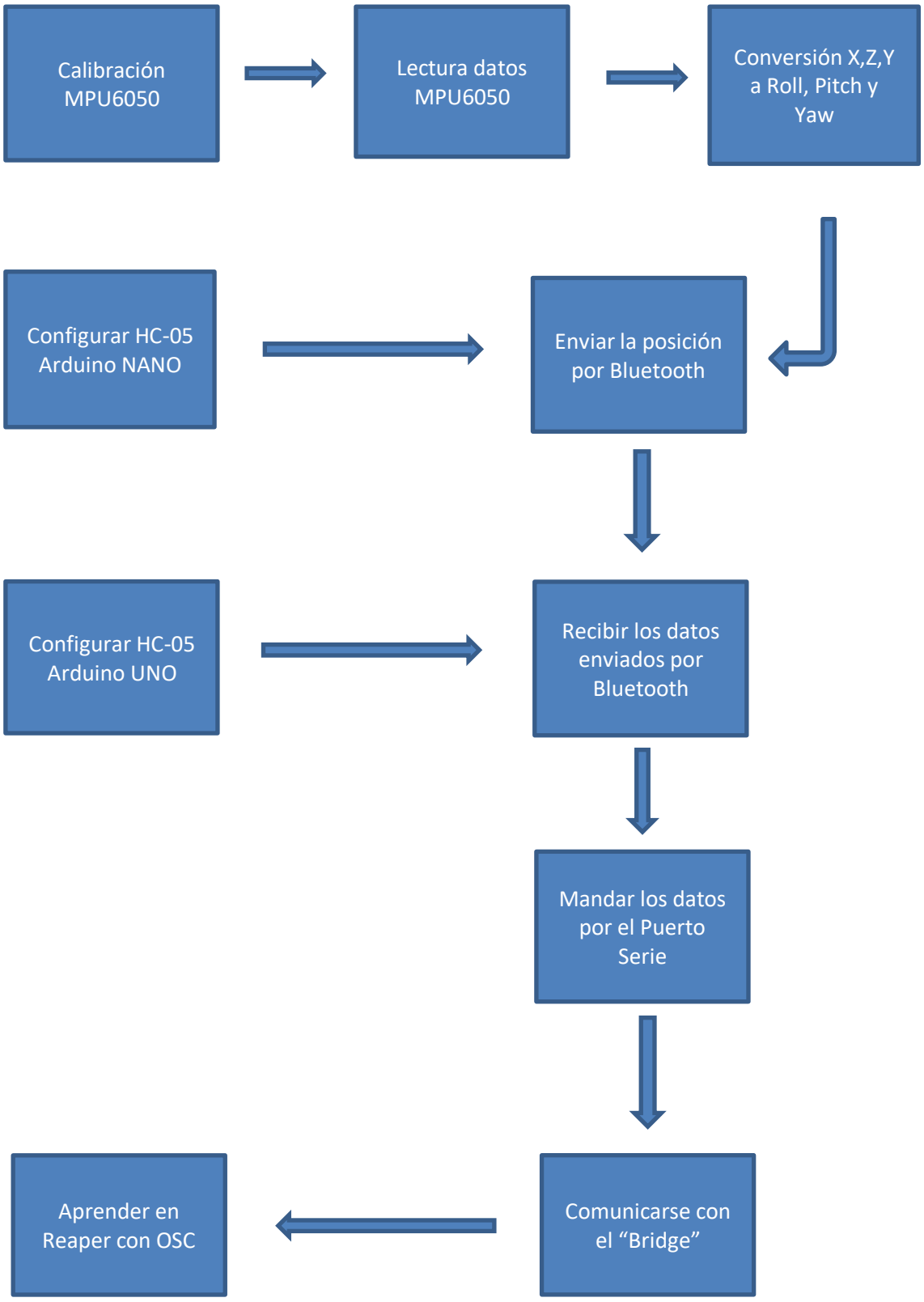


FIGURA 45: SceneRotator (IEM)

Con esta configuración se puede comprobar como cambia la dirección del sonido en función de la posición del oyente que lleva el Head-Tracker. (GitHub - trsonic/nvsonic-head-tracker: Spatial Audio 3DOF Head Tracker (requires Arduino Pro Micro + MPU-9250 / MPU-9150) n.d.)

Para comprender todo el procedimiento se expone el siguiente diagrama de flujo. Los códigos de programación correspondientes a cada uno de ellos se encuentran en los anexos.


Calibración MPU6050
Lectura datos MPU6050
Conversión X,Z,Y a Roll, Pitch y Yaw
Configurar HC-05 Arduino NANO
Enviar la posición por Bluetooth
Configurar HC-05 Arduino UNO
Recibir los datos enviados por Bluetooth
Mandar los datos por el Puerto Serie
Comunicarse con el "Bridge"
Aprender en Reaper con OSC

4.5- Diseño final

Para dotar al proyecto final de un diseño más compacto y estético se ha optado por fabricar una caja con impresión 3D donde se introducen todos los componentes electrónicos del Head-Tracker. Consta de una tapa para asegurar la posición de los componentes. El diseño de las dos partes ha sido realizado en Fusion 360 y Ultimaker Cura. La impresión se ha llevado a cabo en una impresora Creality Ender-3.

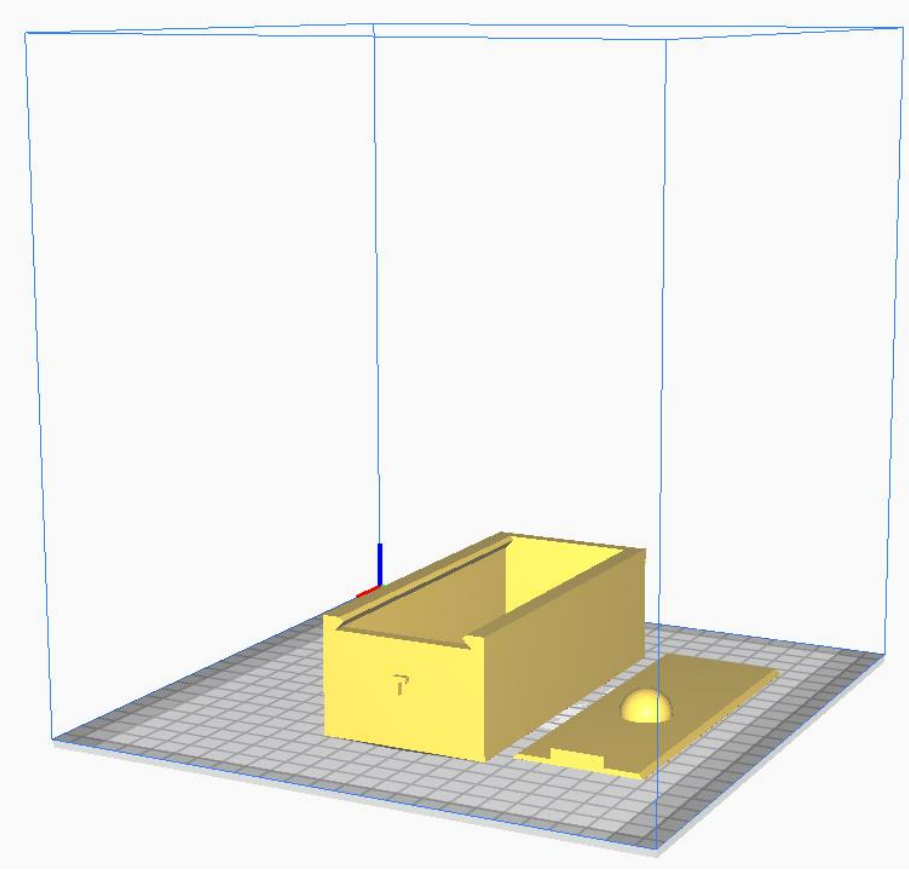


FIGURA 46: Diseño para la impresión 3D

Uno de los problemas a solucionar es la alimentación de todo el dispositivo, para ello se ha optado por una **power bank** que proporciona la alimentación suficiente y da al usuario la posibilidad de cargarla. Se le han añadido un cable para la alimentación y otro de tierra para reducir el tamaño.

FIGURA 47: Power Bank

Una vez se tienen todos los componentes conectados se introducen en la caja con el acelerómetro en posición horizontal y orientado hacia la derecha del oyente, posición que se ha tomado como referencia para calibrarlo.

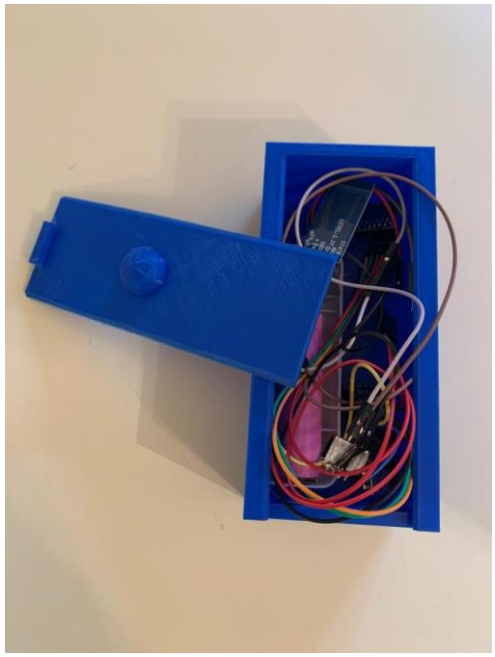


FIGURA 48: Dispositivo con el conjunto 3D

Por último, se añade el conjunto a unos auriculares y se fijan con unas tiras de velcro.

FIGURA 49: Diseño final

CONCLUSIONES

- Se ha desarrollado un sistema de Head-Tracker utilizando Arduino y un acelerómetro MPU6050.
- Se ha conseguido establecer una comunicación Bluetooth entre las dos partes que componen el dispositivo para que el dispositivo sea inalámbrico con dos HC-05.
- Se ha establecido una comunicación del Head-Tracker con Reaper utilizando OSC.
- Se ha logrado rotar la escena en función de la posición del oyente.
- Se ha construido un soporte con impresión 3D para colocar el dispositivo en los auriculares.
- Uno de los objetivos principales del proyecto era que el dispositivo fuera low-cost y se ha conseguido con un presupuesto estimado de entre 40€ - 80€ en función de realizar la compra de los componentes en los sitios oficiales.
- Además, se ha profundizado en conocimientos vistos en la carrera como Virtual Surround, Ambisonic o el audio de próxima generación.

LÍNEAS FUTURAS

El presente proyecto fin de grado tiene algunas posibilidades de cara a mejoras.

- Soldar los diferentes componentes para que el dispositivo sea más compacto.
- Reducir el tiempo de latencia de las lecturas sin perder precisión en la estimación de los ángulos.
- Reducir el tamaño del dispositivo final. Para ello existen diferentes soluciones. Se podría diseñar con un ESP32 más pequeño y que incorpore Bluetooth. La batería se puede sustituir por una más pequeña LIPO de 3.3V. Otra opción sería realizar un PCB personalizado.

BIBLIOGRAFÍA

Algazi, Ralph, Carlos Avendano, Richard O Duda, and V Ralph Algazi. 2001. "109 PACS Numbers: 43.66.Qp, 43.66.Pn DWG." *J. Acoust. Soc. Am*. https://escholarship.org/uc/item/7rf421w9 (May 12, 2020).

"Arduino - Home." https://www.arduino.cc/ (May 14, 2020).

"Configuración Del Módulo Bluetooth HC-05 Usando Comandos AT." https://naylampmechatronics.com/blog/24_configuracion-del-modulo-bluetooth-hc-05-usa.html (May 20, 2020).

"GitHub - Trsonic/Nvsonic-Head-Tracker: Spatial Audio 3DOF Head Tracker (Requires Arduino Pro Micro + MPU-9250 / MPU-9150)." https://github.com/trsonic/nvsonic-head-tracker (May 21, 2020).

"GRASSound&Vibration." http://kemar.us/ (May 12, 2020).

"HRTF Data – The CIPIC Interface Laboratory Home Page." https://www.ece.ucdavis.edu/cipic/spatial-sound/hrtf-data/ (May 12, 2020).

"Introducción Al Diseño e Impresión En 3D (Agustín 'Flowalistik' Arroyo). Curso Online | Domestika." https://www.domestika.org/es/courses/833-introduccion-al-diseno-e-impresion-en-3d (May 14, 2020).

"LISTEN HRTF DATABASE." http://recherche.ircam.fr/equipes/salles/listen/index.html (May 12, 2020).

Majdak, Piotr et al. 2013. "Spatially Oriented Format for Acoustics: A Data Exchange Format Representing Head-Related Transfer Functions." *134th Audio Engineering Society Convention 2013* (May): 262–72.

Møller, Henrik. 1992. "Fundamentals of Binaural Technology." *Applied Acoustics* 36(3–4): 171–218.

"NEUMANN." https://en-de.neumann.com/ku-100 (May 12, 2020).

"Nx – Virtual Mix Room over Headphones | Waves." https://www.waves.com/plugins/nx?gclid=CjwKCAjwqpP2BRBTEiwAfpiD-xX-wIhUeDS49nMrCJjG1B_RHX8lPAndF4rntGiMQ8PbLF8DB_5cNxoCNNcQAvD_BwE#introducing-nx-virtual-mix-room (May 20, 2020).

Olivieri, Ferdinando, Nils Peters, and Deep Sen. 2019. *Scene-Based Audio and Higher Order Ambisonics: A Technology Overview and Application to Next-Generation Audio, VR and 360° Video*. https://tech.ebu.ch/publications (May 15, 2020).

Park, Thomas J., Achim Klug, Michael Holinstat, and Benedikt Grothe. 2004. "Interaural Level Difference Processing in the Lateral Superior Olive and the Inferior Colliculus." *Journal of Neurophysiology* 92(1): 289–301. https://www.physiology.org/doi/10.1152/jn.00961.2003 (May 12, 2020).

"Physics Today On The Web - Cover Story." https://web.archive.org/web/20010306145145/http://www.aip.org/pt/nov99/locsound.html (May 12, 2020).

"SADIE | Spatial Audio For Domestic Interactive Entertainment." https://www.york.ac.uk/sadie-project/database.html (May 12, 2020).

"Tutorial MPU6050, Acelerómetro y Giroscopio." https://naylampmechatronics.com/blog/45_Tutorial-MPU6050-Acelerómetro-y-Giroscopio.html (May 20, 2020).

UNIVERSIDAD POLITÉCNICA DE MADRID ESCUELA UNIVERSITARIA DE INGENIERÍA TÉCNICA DE TELECOMUNICACIÓN PROYECTO FIN DE CARRERA.

Zotter, Franz, and Matthias Frank. 2019. "XY, MS, and First-Order Ambisonics." In Springer, Cham, 1–22.

ANEXOS

Anexo II. Código programación Arduino NANO

```
#include "freeram.h"
#include "mpu.h"
#include "I2Cdev.h"
#include <SoftwareSerial.h>   // Incluimos todas las librerias
necesarias

SoftwareSerial BT(2,3);    // Definimos los pines RX y TX del Arduino
conectados al Bluetooth

#define DISPLAY_INTERVAL  50  // Fijamomos el tiempo entre cada
lectura (ms)

void setup() {

    BT.begin(38400); // Inicializamos el puerto serie BT (Modo AT 2)
    Fastwire::setup(400,0);
    Serial.begin(115200);
    mympu_open(200);
}

unsigned long lastDisplay = 0;

void loop() {
    unsigned long now = millis();
    mympu_update();     // Leemos los datos del acelerometro

    if ((now - lastDisplay) >= DISPLAY_INTERVAL)
    {
      char imu_data[64];
      char qW[8], qX[8], qY[8], qZ[8];

      dtostrf(mympu.qW, 7, 4, qW); // Convierte los numeros en cadenas
      dtostrf(mympu.qX, 7, 4, qX);
      dtostrf(mympu.qY, 7, 4, qY);
      dtostrf(mympu.qZ, 7, 4, qZ);

      strcpy(imu_data,qW);       // Concatenamos las cadenas
      strcat(imu_data,",");
      strcat(imu_data,qX);
      strcat(imu_data,",");
      strcat(imu_data,qY);
      strcat(imu_data,",");
      strcat(imu_data,qZ);

      BT.write(imu_data);   // Mandamos los datos por el Bluetooth
      BT.write(";");
      lastDisplay = now;
    }
```

Anexo II. Código programación Arduino UNO

```
#include "freeram.h"
#include "mpu.h"
#include "I2Cdev.h"
#include <SoftwareSerial.h>   // Incluimos todas las librerias
necesaris

SoftwareSerial BT(10,11);    // Definimos los pines RX y TX del
Arduino conectados al Bluetooth

char x;          // Creamos la variable donde guardaremos los datos

void setup()
{
    BT.begin(38400); // Inicializamos el puerto serie BT (Modo AT 2)
    Fastwire::setup(400,0);
    Serial.begin(115200);
    mympu_open(200);
}

void loop()
{
   if(BT.available())    // Si llega un dato por el puerto BT se envía
al monitor serial
    {
      x= BT.read(); // Guardamos la entrada del Bluetooth en x

      Serial.write(x);   // Se manda x por el puerto serie
   }

}
```

Anexo III. Código programación calibración MPU6050

```
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

// La dirección del MPU6050 puede ser 0x68 o 0x69, dependiendo
// del estado de AD0. Si no se especifica, 0x68 estará implicito
MPU6050 sensor;

// Valores RAW (sin procesar) del acelerometro y giroscopio en los
ejes x,y,z
int ax, ay, az;
int gx, gy, gz;

//Variables usadas por el filtro pasa bajos
long f_ax,f_ay, f_az;
int p_ax, p_ay, p_az;
long f_gx,f_gy, f_gz;
int p_gx, p_gy, p_gz;
int counter=0;

//Valor de los offsets
```

```
int ax_o,ay_o,az_o;
int gx_o,gy_o,gz_o;

void setup() {
  Serial.begin(38400);   //Iniciando puerto serial
  Wire.begin();           //Iniciando I2C
  sensor.initialize();    //Iniciando el sensor

  if (sensor.testConnection()) Serial.println("Sensor iniciado
correctamente");

  // Leer los offset los offsets anteriores
  ax_o=sensor.getXAccelOffset();
  ay_o=sensor.getYAccelOffset();
  az_o=sensor.getZAccelOffset();
  gx_o=sensor.getXGyroOffset();
  gy_o=sensor.getYGyroOffset();
  gz_o=sensor.getZGyroOffset();

  Serial.println("Offsets:");
  Serial.print(ax_o); Serial.print("\t");
  Serial.print(ay_o); Serial.print("\t");
  Serial.print(az_o); Serial.print("\t");
  Serial.print(gx_o); Serial.print("\t");
  Serial.print(gy_o); Serial.print("\t");
  Serial.print(gz_o); Serial.print("\t");
  Serial.println("nnEnvie cualquier caracter para empezar la
calibracionnn");
  // Espera un caracter para empezar a calibrar
  while (true){if (Serial.available()) break;}
  Serial.println("Calibrando, no mover IMU");

}

void loop() {
  // Leer las aceleraciones y velocidades angulares
  sensor.getAcceleration(&ax, &ay, &az);
  sensor.getRotation(&gx, &gy, &gz);

  // Filtrar las lecturas
  f_ax = f_ax-(f_ax>>5)+ax;
  p_ax = f_ax>>5;

  f_ay = f_ay-(f_ay>>5)+ay;
  p_ay = f_ay>>5;

  f_az = f_az-(f_az>>5)+az;
  p_az = f_az>>5;

  f_gx = f_gx-(f_gx>>3)+gx;
  p_gx = f_gx>>3;

  f_gy = f_gy-(f_gy>>3)+gy;
  p_gy = f_gy>>3;

  f_gz = f_gz-(f_gz>>3)+gz;
  p_gz = f_gz>>3;

  //Cada 100 lecturas corregir el offset
```

```
  if (counter==100){
    //Mostrar las lecturas separadas por un [tab]
    Serial.print("promedio:"); Serial.print("t");
    Serial.print(p_ax); Serial.print("\t");
    Serial.print(p_ay); Serial.print("\t");
    Serial.print(p_az); Serial.print("\t");
    Serial.print(p_gx); Serial.print("\t");
    Serial.print(p_gy); Serial.print("\t");
    Serial.println(p_gz);

    //Calibrar el acelerometro a 1g en el eje z (ajustar el offset)
    if (p_ax>0) ax_o--;
    else {ax_o++;}
    if (p_ay>0) ay_o--;
    else {ay_o++;}
    if (p_az-16384>0) az_o--;
    else {az_o++;}

    sensor.setXAccelOffset(ax_o);
    sensor.setYAccelOffset(ay_o);
    sensor.setZAccelOffset(az_o);

    //Calibrar el giroscopio a 0°/s en todos los ejes (ajustar el
offset)
    if (p_gx>0) gx_o--;
    else {gx_o++;}
    if (p_gy>0) gy_o--;
    else {gy_o++;}
    if (p_gz>0) gz_o--;
    else {gz_o++;}

    sensor.setXGyroOffset(gx_o);
    sensor.setYGyroOffset(gy_o);
    sensor.setZGyroOffset(gz_o);

    counter=0;
  }
  counter++;
}
```

Anexo IV. Código programación configuración HC-05

```
#include <SoftwareSerial.h>   // Incluimos la
librería  SoftwareSerial

SoftwareSerial BT(10,11);    // Definimos los pines RX y TX del
Arduino conectados al Bluetooth

void setup()
{
  BT.begin(115200);       // Inicializamos el puerto serie BT (Para
Modo AT 2)
  Serial.begin(11520);   // Inicializamos  el puerto serie
}

void loop()
{
  if(BT.available())     // Si llega un dato por el puerto BT se envía
al monitor serial
  {
    Serial.write(BT.read());
  }

  if(Serial.available())  // Si llega un dato por el monitor serial se
envía al puerto BT
  {
    BT.write(Serial.read());
  }
}
```